Universidad Autónoma de Madrid
Escuela politécnica superior

Proyecto fin de carrera

RECONOCIMIENTO DE IDIOMA EN VOZ ESPONTANEA MEDIANTE ENTRENAMIENTO DISCRIMINATIVO MMI

Ingeniería de Telecomunicación

Juan Bonillo Molina
Junio de 2011

RECONOCIMIENTO DE IDIOMA EN VOZ ESPONTANEA MEDIANTE ENTRENAMIENTO DISCRIMINATIVO MMI

AUTOR: Juan Bonillo Molina
TUTOR: Joaquín Gonzalez Rodriguez

Área de Tratamiento de Voz y Señales
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2011

Resumen

En el presente proyecto se aborda la problemática de reconocimiento idioma a través del empleo del paradigma de entrenamiento discriminativo MMI (Maximum Mutual Information), ya utilizado en otras tecnologías del habla, pero apenas si investigado en el campo del reconocimiento de idioma, donde ha demostrado producir resultados comparables a otros sistemas del estado del arte.

El proyecto comienza haciendo un recorrido por las distintas técnicas empleadas para construir reconocedores de idioma, desde las ya clásicas GMM (Modelos de Mezclas Gausianas) y PRLM, hasta las mas nuevas y poderosas SVM (Maquinas de Vectores Soporte) y Eigenchannel Compensation (Factor Analysis) empleadas en las últimas evaluaciones NIST de idioma. Se describe también el proceso de extracción de características de la señal de voz junto con las técnicas existentes (algunas de ellas implementadas durante el proyecto) para mitigar los efectos adversos producidos por las variaciones del canal y el ruido.

En la parte experimental del proyecto, se han desarrollado un total de 5 reconocedores de idioma diferentes: (sistemas GMM-ML,GMM-MMI,GMM-UBM,AGMM-SVM, y Eignenchannel Compensation), poniendo especial énfasis en el sistema central del proyecto, el sistema GMM-MMI. Dicho sistema, ha sido sometido a las pruebas de evaluación de NIST (National Institute of Standards and Technology) LRE 2003, NIST LRE 2005 y a la evaluación nacional de idioma ALBAYZIN LV 2008. Para esta última evaluación también se ha desarrollado un sistema discriminativo, el sistema denominado AGMM-SVM, que ha llegado a superar en precisión incluso al sistema GMM-MMI central del proyecto. Estos dos sistemas han sido fusionados junto con otros desarrollados por el grupo ATVS, para participar en la evaluación de Albayzin de verificación de la lengua. El sistema resultante ha sido además el ganador de dicha evaluación.

Finalmente, se extraen conclusiones y se proponen nuevas líneas de trabajo futuras.

Abstract

In this project, language recognition problem is addressed using a novel training paradigm, in particular discriminative training MMI (Maximum Mutual Information), already used in other speech technologies, but little explored in the research area of language recognition, where it has been shown to produce results comparable to other state of the art systems.

The project begins by showing the different techniques used to build language recognizers from the classical GMM (Gaussian Mixture Models) and PRLM, to the most powerful new SVM (Support Vector Machines) and Eigenchannel Compensation (Factor Analysis) used in the last NIST Language Recognition Evaluation. It also describes the feature extraction process of the voice signal together with existing techniques (some of them implemented during the project) used to remove the adverse effects caused by channel variability and noise.

During the experimental part of the project, a total of 5 different language recognizers have been developed (GMM-ML, MMI-GMM, GMM-UBM, AGMM-SVM, Eignenchannel Compensation), with special emphasis on the central system of the project, GMM-MMI system. This system has been tested in the evaluations of NIST (National Institute of Standards and Technology) LRE 2003, NIST LRE 2005 and ALBAYZIN LV 2008. For the evaluation of Albayzin another discriminative system has also been developed, called AGMM-SVM. This has become even more accurate than GMM-MMI system's central project.These two systems have been integrated with others developed by ATVS and submitted to the 2008 ALBAYZIN Language Verification Evaluation. The resulting system was the winner.

Finally, conclusions are drawn, and new lines of work are proposed.

Palabras Clave

Reconocimiento (identificación) de idioma, verificación (detección) de idioma, entrenamiento generativo, entrenamiento discriminativo, criterio EM, criterio ML, criterio MMI, QuickProp, Gradient Descent, Support Vector Machine (SVM) , supervector, Gaussian Mixture Model (GMM), Factor Análisis, Phones-SVM ,PRLM, PPRLM,PPR, RASTA, CMS, CMVN, Feature Warping, T-NORM, Z-NORM, Fusión, FAR, FRR, EER, Curva DET ,curva ROC, HTK, SPHINK, parámetros MFCC, parámetros SDC, UBM, MAP , CallFriend, OGI, Albayzin LV, NIST LRE, Eigenchannel Compensation.

Agradecimientos

Quiero dar mi más sincero agradecimiento, en primer lugar, a mi tutor de proyecto, Joaquín González, por darme la posibilidad de realizar el presente proyecto en el grupo de investigación del ATVS, puntero en la investigación y desarrollo de sistemas biométricos. Igualmente a todos los miembros del grupo sin cuya valiosa ayuda este proyecto difícilmente habría visto la luz. Finalmente, a mis padres, amigos y compañeros de carrera, por el apoyo incondicional que me han mostrado durante el desarrollo del proyecto y a lo largo de toda la carrera.

Juan Bonillo Molina
Junio de 2011

Esas largas cadenas de trabadas razones muy simples y fáciles, que los geómetras acostumbran a emplear para llegar a sus más difíciles demostraciones, me habían dado ocasión para imaginar que todas las cosas que entran en la esfera del conocimiento humano se encadenan de la misma manera; de suerte que, con sólo abstenerse de admitir como verdadera ninguna que no lo fuera y de guardar siempre el orden necesario para deducir las unas de las otras, no puede haber ninguna, por lejos que se halle situada o por oculta que esté, que no se llegue a alcanzar y descubrir.

René Descartes, en su Discurso del Método

Este proyecto ha sido realizado en el Área de Tratamiento de Voz y Señales (ATVS) en la Escuela Politécnica superior de la Universidad Autónoma de Madrid.

Índice general

Índice de figuras

Índice de tablas

1

Introducción

1.1. Motivación del proyecto

En un mundo por momentos más globalizado, en donde las fronteras entre países tienden a desaparecer, y las necesidades de comunicación entre personas de distintas lenguas se multiplican, el reconocimiento de idioma adquiere cada vez más protagonismo. Sistemas telefónicos multilingües que precisan de la identificación del idioma del hablante para su correcto reconocimiento, sistemas de indexación de contenidos multimedia por idioma, enrutamiento de llamadas, etc, son los grandes beneficiados de la investigación y avances realizados en este campo durante las últimas dos décadas.

En la actualidad los sistemas del estado del arte de reconocimiento de la lengua ya cuentan con la suficiente madurez como para ser usados con plena seguridad en entornos reales, pero siguen siendo incapaces de superar la precisión que logran humanos entrenados, especialmente cuando operan en condiciones de canal adversas como el ruido presente en las comunicaciones.

El desarrollo de sistemas más precisos y robustos es uno de los mayores retos que mantiene ocupados a los investigadores de este campo. Afortunadamente, el reconocimiento de idioma no está solo ante estos problemas y continuamente se ve ayudado por sistemas análogos de reconocimiento de patrones que hacen uso de la señal hablada, como son el reconocimiento automático de voz y de locutor (especialmente de esta última). Precisamente la técnica que se ha implementado en el presente proyecto, el entrenamiento discriminativo MMI, tiene su origen en el campo del reconocimiento automático de voz, y solo recientemente se ha visto su utilidad en reconocimiento de idioma.

El presente proyecto nace, pues, de la necesidad de explorar nuevos paradigmas de entrenamiento frente a las técnicas clásicas que se han venido empleando sin grandes cambios desde que en 1992 fuera liberada la base de datos OGI [20], momento en el que el problema de reconocimiento de idioma empezó a suscitar un interés creciente en los grupos de investigación de voz, principalmente porque la base de datos OGI les daba la oportunidad de comparar el rendimiento de sus sistemas. Prueba de ello fue la realización de la primera evaluación de idioma de NIST en 1993, utilizando por supuesto esta última base de datos (con una porción no pública de la misma, esto es, a través de un proceso de evaluación ciega).

1.2. Objetivos y enfoque

El objetivo primario del proyecto ha sido investigar el uso del criterio discriminativo MMI para el desarrollo de un reconocedor de idioma en voz espontánea.

Dicho reconocedor de idioma, denominado en adelante GMM-MMI, deberá, dado un segmento de audio (obtenido de una llamada telefónica) determinar el idioma más probable contenido en dicho segmento (supuesto que éste opere en modo detección o identificación) o bien verificar si un idioma determinado está presente o no en el mismo (si el sistema opera en modo verificación).

Para completar con éxito dicho objetivo, en el capitulo 2 se estudia el estado actual de las tecnologías de reconocimiento de idioma, y cual es el papel que puede jugar el entrenamiento discriminativo, en particular el basado en el criterio de máxima información mutua o MMI, en la contribución a la mejora del estado del arte del reconocimiento de idioma. Se describen primeramente los sistemas acústicos por estar el sistema GMM-MMI basado en ellos y a continuación los sistemas fonéticos basados en PRLM (Phone Recognition followed by Language Modelling), que lideraron el estado del arte desde la evaluación de idioma de NIST 1993 hasta la evaluación de idioma NIST 2003, momento en el que fueron superados por los sistemas acústicos (esto obligo al desarrollo de sistemas fonéticos mas competitivos como el sistema Phones-SVM también descrito en este capitulo).

En el capitulo 3 se verá el procedimiento de extracción de características de la señal de voz, para convertir una larga secuencia de muestras de audio en un vector de pocas componentes que los reconocedores puedan emplear sin problemas. También se describen algunos métodos para eliminar efectos lineales indeseables para él reconocimiento, introducidos por la respuesta en frecuencia del canal de transmisión (entre ellos Feature Warping, CMS y RASTA).

El capitulo 4 esta dedicado por completo al estudio del criterio MMI para su uso en reconocimiento de idioma. Después de definir la función objetivo MMI, se describen matemáticamente tres de los algoritmos implementados para su optimización, y, en particular, el algoritmo Baum-Welch-Extendido que fue el que se termino utilizando en los experimentos. Se termina el capitulo con un ejemplo de aplicación gráfico del criterio MMI, donde se analizarán las bondades del mismo frente al entrenamiento generativo del criterio ML.

En el capitulo 5, se analizan las herramientas de software, hardware y entorno de trabajo empleadas para la elaboración del proyecto. Gran parte del software utilizado ha sido de producción propia (para lo que se han necesitado más de 10000 líneas de código). Se ha prestado especial atención para que todo el software desarrollado pueda ser usado en el entorno de desarrollo del grupo ATVS (incluyendo un manual de usuario de los sistemas implementados durante el proyecto). También describimos en este capitulo como se han diseñado dos de los sistemas presentados a la evaluación de idioma de Albayzin de 2008, el sistema primario GMM-MMI y el sistema de contraste AGMM-SVM.

En el capitulo 6 del proyecto se abordan los métodos existentes para la medida de rendimiento de los sistemas desarrollados. En particular se describen los protocolos de evaluación seguidos en NIST y Albayzin. También veremos las bases de datos que se han utilizado para el desarrollo y evaluación de los sistemas.

El Capitulo 8, sin duda el más importante de todos, describe los experimentos llevados a cabo para el desarrollo del sistema GMM-MMI, además de los sistemas Eigenchannel Compensation y AGMM-SVM (este último desarrollado para la evaluación de Albayzin de idioma) y su evaluación en NIST LRE 2003, NIST LRE 2005 y Albayzin LV 2008.

En el capitulo 9, se extraerán una serie de conclusiones y se proporcionarán una serie de directrices o claves de éxito, sobre los sistemas desarrollados.

Finalmente, se analizarán posibles líneas de trabajo futuro para la construcción de sistemas de reconocimiento de idioma más precisos haciendo uso de la aportación realizada con el proyecto.

2

Estudio del estado del arte

2.1. Introducción al reconocimiento de idioma

En este primer capitulo del proyecto, veremos en el estado en que se encuentra la tecnología de reconocimiento de idioma. Para ello comenzaremos primeramente viendo en que aplicaciones resulta útil disponer de esta tecnología.

2.1.1. Aplicaciones

Los sistemas de reconocimiento idiomático se usan principalmente en entornos telefónicos multilingües tanto automáticos como manuales (donde un operador humano recibe la llamada).

En tales sistemas donde la única forma de comunicación del usuario con el sistema es a través de la voz, el sistema debe ser capaz de determinar la identidad del idioma del usuario a través de los comandos de éste. Estos comandos pueden ser reconocidos de dos formas. La primera de ellas, durante la realización del comando mismo y la segunda antes de que el comando tenga lugar. La primera aproximación requeriría disponer de muchos sistemas de reconocimiento de voz en paralelo. En aplicaciones complejas donde el número de idiomas es considerable, esto redundaría en un coste computacional prohibitivo, haciendo esta aproximación imposible. La segunda aproximación, haría uso de reconocedores de idioma para determinar la lengua o lenguas mas probables habladas por el usuario, y, ahora si, emplearíamos los reconocedores de voz adecuados (idealmente uno solo si el sistema de identificación fuera perfecto).

En sistemas telefónicos multilingües no automáticos (como muchos call-center y algunos teléfonos de servicios de emergencia) se emplean reconocedores de idioma para enrutar la llamada a un operador humano que entienda el idioma del usuario. De esta forma se automatiza el proceso y se consigue prestar el servicio mucho más rápido que de forma manual. Enrutar la llamada al operador adecuado (que hable el idioma del usuario) lo más rápidamente posible en servicios de emergencia (bomberos, policía, urgencias médicas) puede llegar a salvar vidas humanas.

Otro campo de aplicación y que esta ganando más auge con los años debido a la creciente explosión de información audiovisual (principalmente de internet, radio, y televisión) es en la clasificación y recuperación de contenidos de audio por idioma.

Por último, utilizando reconocedores de idioma en sistemas de traducción simultanea voz a voz entre varios interlocutores, evitaríamos el tener que configurarlos.

En la siguiente sección describiremos formalmente el problema de reconocimiento de la lengua que subyace a todas estas aplicaciones.

2.2. Formulación matemática del problema de reconocimiento de idioma

Resumimos a continuación una formulación rigurosa del problema de reconocimiento de idioma que como veremos a continuación puede englobarse dentro del marco de reconocimiento de patrones clásico Bayesiano.

Dada una secuencia de segmentos acústicos (obtenidos de un fichero de audio) y que representaremos por $O = \{o_1, o_2, o_3, ..., o_n\}$ correspondientes a uno de entre M idiomas posibles $L = \{L_1, L_2, L_3, ..., L_M\}$, la tarea de identificación , consistirá en determinar el idioma L^* al que con más probabilidad pertenecerá dicha secuencia de segmentos acústicos, esto es:

$$L^* = \arg \max_i P(L_i|O), \tag{2.1}$$

siendo $P(L_i|O)$ la probabilidad a posteriori del lenguaje L_i.Suponiendo que efectivamente el segmento de entrada O pertenezca a una de las M clases (idiomas) $L_i, 1 \leq i \leq M$, el problema de identificación de idioma se reducirá a un problema típico de clasificación de patrones. El objetivo principal de dicho problema de clasificación de patrones será decidir a que clase pertenecerá el segmento de audio O.

Si aplicamos la regla de Bayes, la expresión (2.1) se puede poner de la forma:

$$P(L_i|O) = \frac{P(O|L_i)P(L_i)}{P(O)} \tag{2.2}$$

En la expresión de arriba podemos eliminar el término del denominador $P(O)$ puesto que no depende del índice i, y por consiguiente, no influye en la decisión de clasificación. De este modo el problema queda reducido a maximizar la función de densidad de probabilidad conjunta $P(O|L_i)P(L_i)$. De acuerdo a la expresión (2.1), deberemos elegir el idioma (o clase) L^* para el cual la probabilidad a posteriori $P(L_i|O)$ sea máxima para el segmento O. Alternativamente, podemos implementar la misma expresión por:

$$L^* = \arg \max_i P(O|L_i)P(L_i), \tag{2.3}$$

donde $P(O|L_i)$ representa la función de densidad de probabilidad del segmento O con respecto al idioma L_i y $P(L_i)$ denota la probabilidad a priori del idioma L_i que por lo general se considera equiprobable para todas las clases. Finalmente, la expresión (2.3) queda como:

$$L^* = \arg \max_i P(O|L_i) \tag{2.4}$$

Por tanto el problema de identificación de idioma consistirá en estimar convenientemente la función de densidad de probabilidad $P(O|L_i)$. Tal como se ha planteado el problema, el sistema trabajaría en modo identificación, modo que será descrito con mas detalle en el siguiente apartado.

2.2.1. Modos de operación

Los sistemas de reconocimiento de idioma pueden trabajar de dos modos distintos dependiendo de las necesidades del problema, a saber:

- ***Modo identificación (comparación 1:N):*** la señal de audio desconocida se enfrenta a los N modelos de idioma de nuestro sistema. De esta forma se obtienen N puntuaciones (una por modelo). Las puntuaciones serán tanto mayores (mas positivas) cuanto mayor sea la probabilidad de que el segmento de voz pertenezca al modelo de idioma correspondiente. La salida del sistema será el idioma cuyo modelo obtenga la máxima puntuación. Por lo general se asume que la señal de audio procede de un conjunto de idiomas conocidos, por esta razón se dice que los sistemas de identificación generalmente trabajan en conjunto cerrado. En la figura de a continuación, se ilustra el esquema básico de un sistema de identificación de idioma:

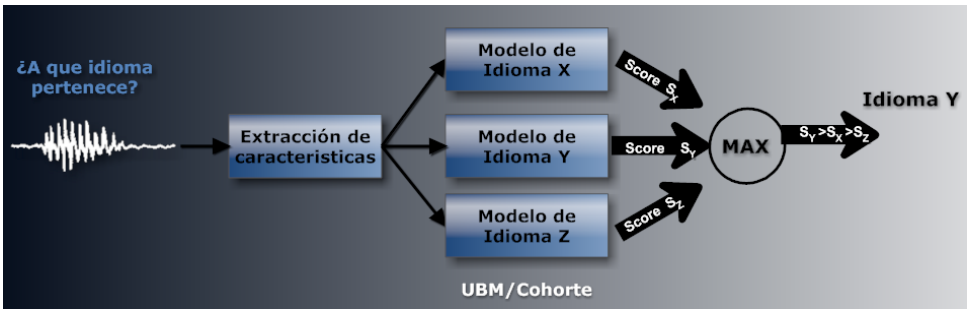

Figura 2.1: Modo identificación.

- ***Modo verificación o detección (comparación 1:1):*** el objetivo es determinar si un segmento de audio contiene un determinado idioma. Para ello el segmento se enfrenta al modelo de idioma que se desea comprobar y a un modelo universal que representa al resto de idiomas posibles. La diferencia de las puntuaciones de los dos modelos se compara con un umbral, respondiendo el sistema con un si/no en función de si la puntuación esta por encima o debajo de este umbral, respectivamente. Debido a que generalmente se asume que la señal de audio puede contener cualquier idioma, un sistema de verificación o detección trabajará en conjunto abierto. Un sistema de verificación puede verse como una caso particular de un sistema de identificación cuando a este último se le agrega un modelo que represente la alternativa a todos los idiomas conocidos por el sistema. Las evaluaciones NIST de idioma emplean este modo de operación. En la siguiente figura se ilustra el esquema básico de un sistema de verificación de idioma:

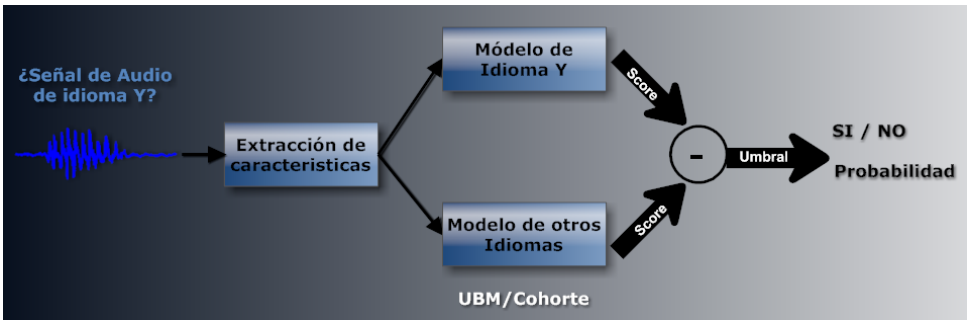

Figura 2.2: Modo verificación.

- ***Modo registro:*** este modo es previo a los dos anteriores. Durante el mismo el audio de los idiomas es procesado para obtener los vectores de características usados por los sistemas de verificación o identificación. De esta forma se crea un registro o base de datos para el entrenamiento y evaluación de los sistemas.

2.2.2. Niveles de información

La información relativa a la identidad del idioma contenida en la señal hablada puede a efectos prácticos considerarse que esta repartida en diversos niveles. La información de cada nivel pueden ser aislada para construir sistemas independientes basándonos solo en uno de ellos. En general, podemos distinguir 6 grandes niveles en función del tipo y la cantidad de información idiomática que contienen (otra cosa distinta es que seamos capaces de extraerla por completo). Se describen estos 6 niveles en orden de menos a más información:

- ***Nivel acústico:*** la información de la señal de voz en si misma $x(t)$, que es en realidad una onda de presión **acústica** que se propaga longitudinalmente a través de las moléculas del aire, se representa de una forma compacta en el dominio cepstral. Normalmente en idioma se emplean los vectores cepstrales MFCC obtenidos en una ventana de la señal de voz de unas decenas de milisegundos (típicamente entre 20 y 30) debido a las características pseudo-periódicas de esta a corto plazo. Estos vectores, que en si mismo representan de forma compacta las características espectrales de la señal de voz a corto plazo, están estrechamente relacionados con el proceso de articulación de sonidos en el tracto vocal y el análisis del modelo de producción de voz, como se verá en posteriores secciones. Debido a que estos vectores cepstrales presentan diferentes patrones estadísticos para cada idioma, es posible modelarlos para su uso en la identificación del idioma.

- ***Nivel prosódico:*** los idiomas tienen distintos patrones prosódicos. Desde el punto de vista perceptual estos patrones prosódicos son la entonación, intensidad (volumen) y ritmo del mensaje hablado y, que a nivel físico de la señal de voz, se corresponden con la frecuencia fundamental (frecuencia a la que vibran las cuerdas vocales), energía y duración de los fonemas, respectivamente. Los sistemas que trabajen a este nivel deberán extraer, con más o menos acierto, estos parámetros prosódicos, de los que la frecuencia fundamental es el más relevante, y posteriormente deberán realizar sobre los mismos algún tipo de modelado estadístico para poder realizar finalmente la clasificación idiomática.

- ***Nivel fonético:*** aunque los seres humanos somos capaces de producir un número prácticamente ilimitado de sonidos, cada idioma tendrá un número reducido de sonidos característicos (fonemas) que los distinguirá del resto. Incluso aunque dos idiomas compartieran el mismo número y tipo de fonemas (cosa harto improbable) es posible que algunos de ellos fueran más probables en un idioma que en otro. El número de fonemas en un idioma puede variar de 15 a 50 con una media de 30.

- ***Nivel fonotáctico:*** no solo el número y tipo de fonemas son diferentes entre idiomas, sino la combinación de los mismos o secuencias de fonemas posibles. Algunas secuencias de fonemas que ocurran frecuentemente en un idioma puede que sean imposibles en otro. Los sistemas que trabajan a este nivel emplean modelos estadísticos de idioma basados en n-gramas (subsecuencia de n fonemas) para realizar el reconocimiento.

- ***Nivel léxico (o morfológico):*** conceptualmente la diferencia más importante entre diferentes idiomas es que cada uno usa un conjunto de palabras distintas. Por consiguiente, un hablante no nativo en inglés, por ejemplo, podrá hablar empleando la prosodia e inventario fonético de su idioma nativo pero se entenderá que esta hablando inglés por el vocabulario empleado.

- ***Nivel sintáctico (o gramatical):*** de modo similar al nivel fonotáctico, cada idioma tendrá unas reglas específicas mediante las que combinar palabras durante el discurso y que contendrán gran cantidad de información útil para identificarlo.

Figura 2.3: Niveles de información relativa a la identidad del idioma en la señal hablada.

Aunque los niveles superiores como léxico y gramatical contienen mucha mas información idiomática que el resto de niveles y suelen ser mas robustos frente a efectos de distorsión del canal, tienen el inconveniente de ser más difíciles de generalizar (se requieren reconocedores de voz para cada idioma a identificar, que a su vez requieren audio transcrito difícil de obtener) y pueden resultar muy costosos computacionalmente. Esto hace que los sistemas basados en estos niveles raramente sean empleados en la práctica. Por el contrario los niveles inferiores como el acústico y prosódico tienen como ventaja que son fáciles de aplicar a cualquier número de idiomas a un coste computacional relativamente bajo, sin embargo adolecen de una precisión inferior y de ser menos robustos frente a variaciones del canal y ruido. Por otro lado los niveles intermedios como el fonotáctico, mantienen un compromiso entre robustez, generalización y coste computacional y son junto con los sistemas acústicos los más empleados en reconocimiento de idioma.

En cualquier caso, es obvio que un buen sistema de reconocimiento idealmente deberá explotar todos los niveles de información anteriores para obtener las mejores tasas de precisión. El problema radicara en como conseguir extraer esta información multinivel, como modelarla matemáticamente para su clasificación y finalmente implementar estos modelos matemáticos en un computador con un coste computacional razonable para la aplicación final (en tiempo de proceso y costes de hardware).

2.3. Historia del reconocimiento de idioma

Los primeros estudios conocidos que tratan de abordar seriamente la problemática del reconocimiento de idioma, se produjeron en los años 70 con los trabajos realizados en el departamento de procesado de señal de Texas Instruments. Entonces el reconocimiento se realizaba analizando la frecuencia de ocurrencia de ciertos sonidos de referencia en diferentes lenguajes. Estos sonidos de referencia se tenían que determinar manualmente, lo que no resultaba nada fácil. Esto a su vez provocaba que el rendimiento se degradase considerablemente al incorporar nuevos idiomas al sistema.

En 1977 Neuberg y House desarrollaron conjuntamente un nuevo sistema basado en HMM's que eran entrenados empleando transcripciones fonéticas obtenidas manualmente y demostrando así que es posible construir excelentes reconocedores de idioma empleando únicamente información fonotáctica.

Los avances mas significativos en la década de los 80 fueron los realizados por Ives y Cimarusti en 1982 y por Foil en 1986. Los primeros desarrollaron un sistema basado en un clasificador polinomial de 100 elementos derivados del análisis LPC (estos elementos incluían, entre otros, los coeficientes de autocorrelación, coeficientes cepstrales, y frecuencias de formantes). Con una precisión del 84 % en 8 idiomas utilizando grabaciones de 3 minutos, este estudio demostró, como ya había ocurrido a nivel fonético, la factibilidad de construir buenos reconocedores de idioma basándose exclusivamente en la información acústica de la señal de voz. Foil por su parte construyó el que se puede considerar el primer sistema de identificación idiomático basado en la prosodia. El sistema consistía en extraer 7 características prosódicas (basadas en el ritmo y la entonación) obtenidas del análisis de la energía y frecuencia fundamental (F_0). Como clasificador empleó un cuantificador vectorial basado en k-means, obteniendo una tasa de acierto del 64 % en tres idiomas. El audio con un SNR de 5 db, provenía de la grabación de emisoras de radio.

En la década de los 90 la construcción y liberación de la base de datos OGI (Oregon Graduate Institute) [20] en 1993 supuso toda una revolución en la investigación del reconocimiento de idioma, al permitir por primera vez que grupos de investigación en diferentes partes del globo pudieran comparar y replicar sus resultados. Hasta entonces todos los resultados se habían obtenido utilizando bases de datos privadas con lo que era imposible replicar los resultados de otros grupos. Los primeros sistemas acústicos y fonéticos actuales (como los GMM y PPRLM) se desarrollaron y evaluaron sobre esta base de datos.

En 1993 se celebra la primera evaluación de idioma organizada por NIST utilizando OGI. Posteriormente, en 1996, el LDC (Linguistic Data Consortium) libera la base de datos Callfriend [31] resolviendo así una de las principales limitaciones de OGI, que era la escasa cantidad de audio con la que entrenar los modelos. Callfriend contiene aproximadamente 10 veces más audio, repartido en 12 idiomas (tres de ellos, inglés, español y chino mandarín, con 2 dialectos). Las evaluaciones de idioma de NIST 1996 y 2003 emplearon una porción no pública de esta base de datos para construir los datos de evaluación. Las evaluaciones de idioma de 2005 y 2007 emplean, sin embargo, una porción del corpus OHSU.

En la actualidad se trabaja en dos frentes para mejorar la precisión de los sistemas. El primero es la construcción de sistemas mas robustos frente a las variaciones provocadas por el canal y el locutor, tanto a nivel de extracción de características como de modelado y, en segundo lugar, en mejores técnicas para fusionar diferentes sistemas con distintos niveles de información (acústicos, prosódicos, fonéticos, etc) con mejores sistemas de modelado. Este proyecto intenta mejorar el primer frente a través de un modelado más eficiente de la información acústica utilizando técnicas discriminativas (tales como MMI y SVM).

2.3.1. Estudios perceptuales en seres humanos

Aunque este proyecto trata exclusivamente del reconocimiento automático de idioma, entendido como tal al que se realiza sin ninguna intervención humana, no esta demás echar un vistazo muy breve al reconocimiento de idioma realizado por seres humanos y los diversos estudios que se han realizado al respecto. Entender los procesos de identificación de la lengua por humanos puede ayudarnos a construir mejores sistemas.

Como suele suceder con otros muchos campos del reconocimiento de patrones (entre los que tal vez reconocimiento facial y de voz son los más conocidos), el ser humano sigue siendo el

sistema de reconocimiento de idioma mas preciso y por consiguiente el modelo a imitar y superar (esperemos que no por mucho tiempo). Con muy poca cantidad de audio (aproximadamente un segundo) el ser humano es capaz de reconocer con precisiones superiores al 90 % segmentos de audio hablados en su idioma nativo y mas aún, es capaz de filtrar la señal de voz en condiciones de ruido de fondo muy adversas y con conversaciones entrelazadas, para quedarse por último con la información relevante. Esto significa, que no es suficiente con construir sistemas precisos, sino también sistemas lo suficientemente robustos para que las condiciones del canal y ruido solo disminuyan dicha precisión dentro de rangos tolerables para la tarea a la que se destinan. El problema de la robustez en los sistemas, es uno de los mayores handicaps a los que se enfrentan la mayoría de los sistemas de reconocimiento de patrones actuales, de los que el reconocimiento de idioma no esta exento.

Describimos a continuación los estudios perceptúales en reconocimiento de idioma mas relevantes que se han realizado durante las últimas dos décadas ordenados por orden de publicación.

2.3.1.1. Experimentos de Muthusamy (1994)

El objetivo principal de este experimento era encontrar las principales pistas y estrategias que los seres humanos empleamos para reconocer distintos idiomas, además de la relevancia de factores tales como la precisión del reconocimiento en función de la longitud de los segmentos de test y el número de idiomas conocidos por el sujeto.

Los idiomas bajo estudio (10 en total) fueron los siguientes: *Mandarin, Inglés, Farsi, Francés, Alemán, Japonés, Coreano, Español, Tamil y Vietnamita*. El audio se obtuvo de la base de datos OGI_TS.

Para el estudio se emplearon 28 participantes. Diez de ellos (4 hombres y 6 mujeres) eran monolingües en inglés. Los 18 restantes (además de hablar y escribir el inglés) tenían como idioma nativo cada uno de los 9 idiomas (2 por cada idioma) distintos del inglés.

El experimento se dividió en dos fases. Durante la primera fase del experimento solo los hablantes nativos monolingües en inglés participaron (10 en total). El procedimiento seguido en esta primera fase fue el siguiente:

1. Para que los sujetos se familiarizaran tanto con el procedimiento como con los idiomas de evaluación, se realizó con los mismos una pequeña sesión de entrenamiento de unos 40 segmentos de audio.
2. Después del paso 1) comenzaba la prueba formal consistente en reconocer 760 segmentos de audio con duraciones de 1, 2,4 y 6 segundos. En total, 19 segmentos por cada idioma y duración.
3. Por cada fichero de test contestado, al sujeto se le mostraba la respuesta correcta con el fin de que este pudiera aprender conforme el experimento avanzaba. Además también se le permitía escuchar dicho fichero de test tantas veces como deseara antes y después de haber contestado (y, por consiguiente, saber el idioma al que pertenecía).
4. Una vez concluida la prueba a cada sujeto se le realizaba una entrevista para conocer las estrategias de identificación que había empleado durante el desarrollo del experimento.

Los resultados de esta primera fase pueden verse en la figura siguiente. El primer cuarto se refiere a los 190 primeros ficheros de test y el último cuarto a los últimos 190 ficheros de los 760 en que consistía la prueba. Como podemos apreciar observando la figura, globalmente se ve un

aprendizaje después de que los sujetos escucharon las tres cuartas partes (190*3) de los ficheros de evaluación.

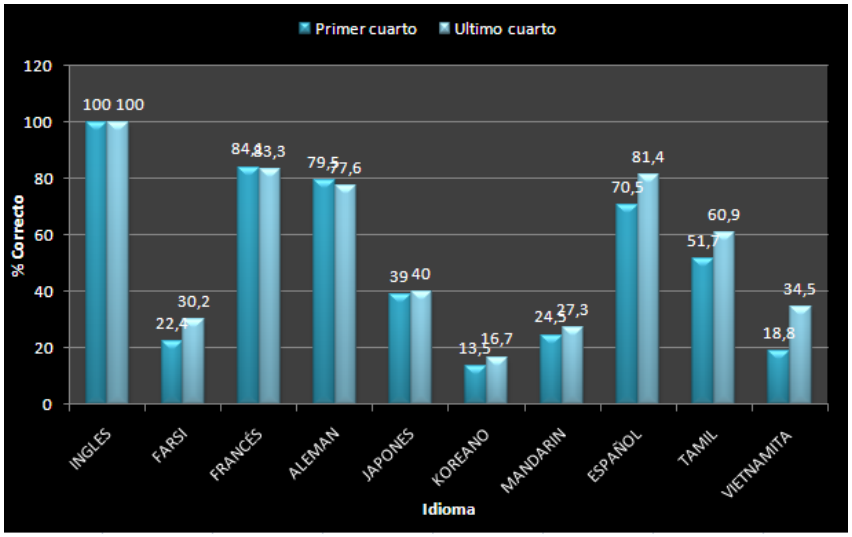

Figura 2.4: Primera fase del experimento: rendimiento medio de los sujetos para los segmentos de 6 segundos en el primer y último cuarto de la prueba.

La segunda fase se llevó a cabo con todos los sujetos. El procedimiento seguido, similar al de la primera fase, fue el siguiente:

1. Para que los sujetos se familiarizaran tanto con el procedimiento de evaluación como con los idiomas, se realizó con los mismos una pequeña sesión de entrenamiento de unos 80 segmentos de audio.

2. Después del paso 1) comenzaba la prueba formal consistente en reconocer 1520 segmentos de audio con duraciones de 1, 2,4 y 6 segundos. En total, 38 segmentos por cada idioma y duración.

3. Por cada fichero de test contestado, al sujeto se le mostraba la respuesta correcta con el fin de que este pudiera aprender conforme el experimento avanzaba. Además también se le permitía escuchar dicho fichero de test tantas veces como deseara antes y después de haber contestado (y por consiguiente saber el idioma al que pertenecía).

4. Una vez concluida la prueba a cada sujeto se le realizaba un cuestionario para conocer las estrategias de identificación que había empleado durante el desarrollo del experimento.

En las siguientes figuras podemos ver los resultados para los hablantes de inglés y para todos los sujetos. El primer cuarto se refiere a los primeros 380 ficheros de test de la prueba y el último cuarto a los últimos 380 ficheros evaluados. Como podemos observar comparando la figura 1 con la 4 se ha incrementado el rendimiento del último cuarto notablemente al aumentar la exposición al audio de entrenamiento. Otro hecho que salta a la vista comparando las figuras, es que los sujetos que tienen el inglés como idioma nativo tienden, en media, a tener un rendimiento ligeramente peor que el resto de sujetos del experimento.

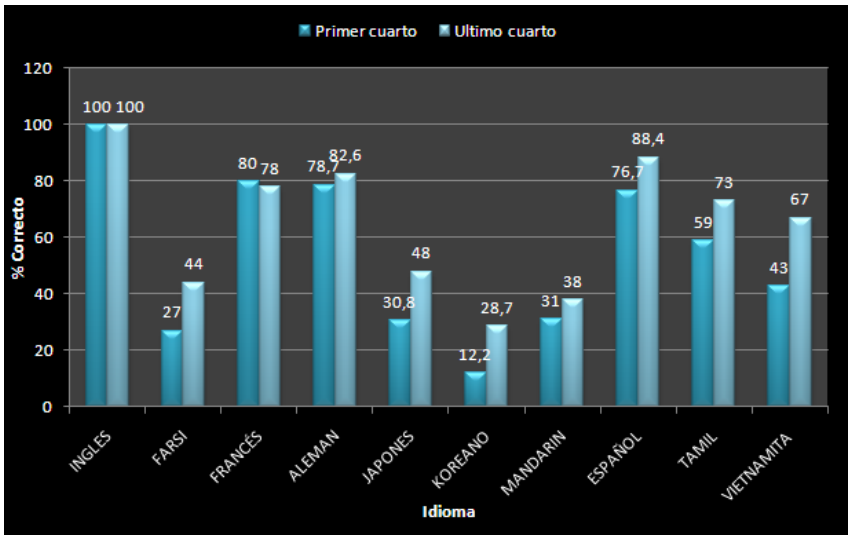

Figura 2.5: Segunda fase del experimento: rendimiento medio de los sujetos para los segmentos de 6 segundos en el primer y último cuarto de la prueba. Sujetos monolingües nativos en inglés.

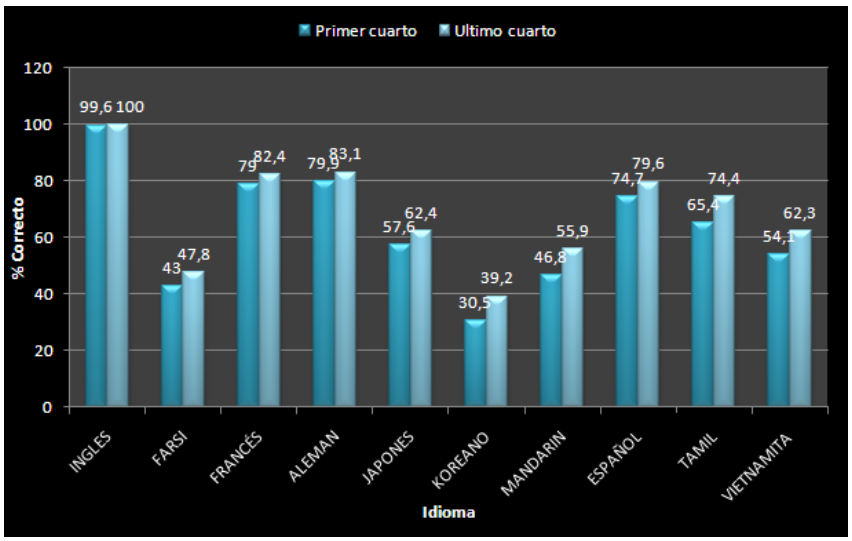

Figura 2.6: Segunda fase del experimento: rendimiento medio de los sujetos para los segmentos de 6 segundos en el primer y último cuarto de la prueba. Todos los sujetos del estudio.

La siguiente figura, resalta el hecho de que sujetos que hablaban mas idiomas, tendían, en media (excluyendo los idiomas que conocen), a reconocer con mayor precisión el resto de idiomas de los segmentos de evaluación.

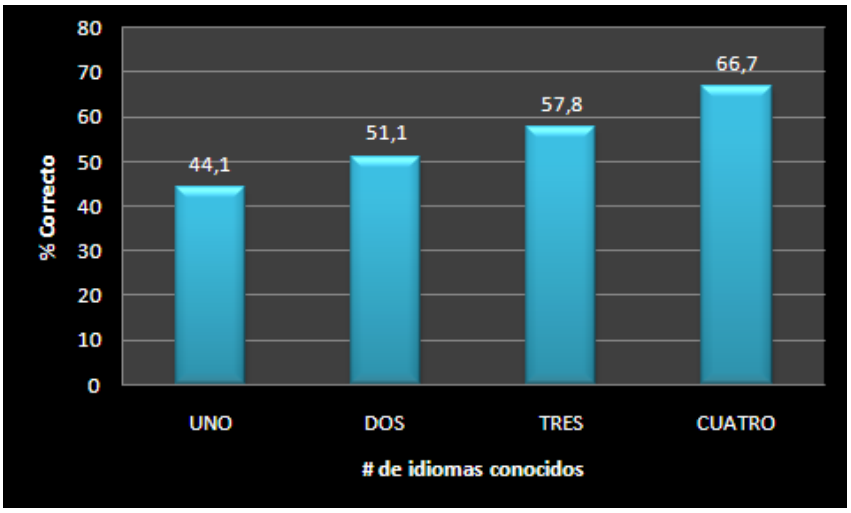

Figura 2.7: Rendimiento medio en función del número total de idiomas que hablan los sujetos.

Como ya se ha mencionado mas arriba, al final del experimento se realizó una entrevista a todos los sujetos para descubrir las estrategias que habían seguido para realizar el reconocimiento, pues bien, estas fueron las conclusiones tras el análisis:

- El francés tiene diversos sonidos nasales características. La mayoría de los sujetos encontraron su entonación distinta.
- Las pausas en japonés son muy bruscas, y tiene ciertas palabras singulares típicas como "watashiwa" y "mashita".
- El alemán es fácilmente reconocible por la palabra "ich", además los sonidos aspirados velares son muy frecuentes. Muchos sujetos lo confundieron con el farsi.
- La mayoría encontró al Mandarín un tono informativo .Una minoría lo confundió con japonés.
- El farsi tiene diversos sonidos aspirados y el fonema /sh/ ocurre con mucha frecuencia.
- El coreano fue con diferencia el idioma más difícil. Muchos de los sujetos buscaban la ocurrencia de la palabra típica "imnida".

Debido a la variedad de sujetos e idiomas se hace difícil sacar conclusiones genéricas, por lo que una aproximación por pares de idiomas sería sin duda más concluyente a la hora de diferenciar los rasgos discriminativos que emplean los sujetos para identificar los idiomas.

2.3.1.2. Experimentos de Navratil (2001)

Como ya hemos mencionado previamente cuando estudiamos los niveles de información contenidos en la señal de voz, el ser humano hace uso de todos los niveles para lograr identificar con una gran tasa de éxito el idioma contenido en una breve locución de unos pocos segundos. Del mismo modo, un buen sistema de reconocimiento idiomático debería hacer uso de todos los niveles, de lo contrario estaría perdiendo información muy valiosa para la tarea de identificación. Estos niveles son el léxico (palabras empleadas), el fonético (sílabas) y el prosódico (acentuación, ritmo y entonación). Desde el punto de vista físico-acústico la prosodia depende de tres parámetros básicos que son el tono, la amplitud y duración de la señal de audio.

La pregunta que nos podríamos hacer es: ¿Que importancia tiene cada nivel en la tarea de identificación de idioma? Y esto es justamente el objetivo primario del experimento de Navratil, medir en seres humanos el papel que juega cada nivel en la precisión de la identificación idiomática lograda.

Con este fin, Navratil utilizo 78 participantes de 12 nacionalidades distintas. El audio empleado para el experimento se corresponde con grabaciones telefónicas de larga distancia en 5 idiomas: Chino, Inglés, Francés, Alemán y Japonés.

A continuación se realizó tres conjuntos de test a los sujetos, uno por cada nivel de información:

- ***Test A (nivel léxico):*** los sujetos tienen que reconocer segmentos de audio de 6 segundos. El audio presentado está sin alterar.
- ***Test B (nivel fonético):*** se extraen la sílabas del audio y se reordenan de forma conveniente para que no contengan información léxica.
- ***Test C (nivel prosódico):*** haciendo uso de la transformada inversa de los parámetros LPC se consigue eliminar la información del tracto vocal y aislar el tono (frecuencia fundamental F_0) y amplitud de la señal.

Los resultados que se obtuvieron se resumen en la siguiente figura. Como era de esperar el nivel léxico resulto ser el más decisivo para identificar el idioma, seguido muy de cerca por el nivel fonético y muy de lejos por el nivel prosódico.

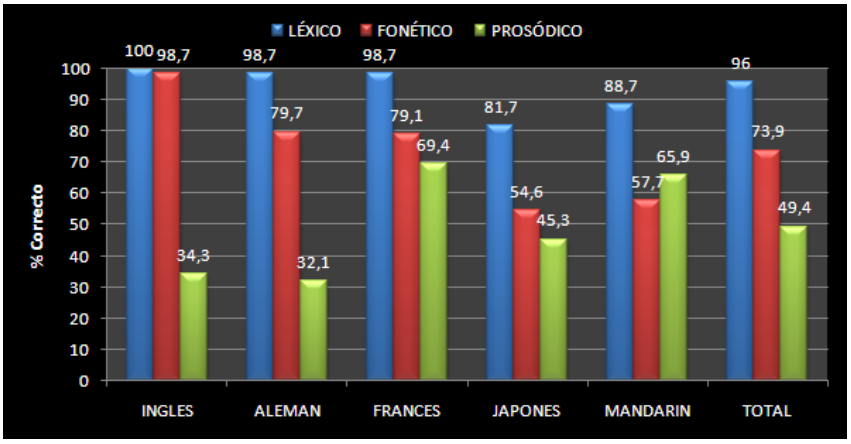

Figura 2.8: Precisión en el reconocimiento en función del nivel de información de idioma empleado por los sujetos del experimento.

2.3.1.3. Experimentos de Maddieson y Vasilescu (2002)

El objetivo primario del experimento era determinar la influencia que podría tener para la identificación de un idioma tanto la familiarización previa como el conocimiento del mismo a nivel lingüístico y así poder discriminar entre un grupo de idiomas objetivo (5 en total) de entre 18 idiomas no objetivo.

Los cinco idiomas objetivos fueron: etíope, rumano, coreano, marroquí, árabe e hindi. Los ficheros de test tenían una duración de 20 segundos. Se formaron dos grupos de participantes: 22 investigadores EE/CS (ingenieros en informática y telecomunicaciones) y 22 lingüistas. Las conclusiones obtenidas tras el experimento fueron las siguientes:

- El conocimiento y familiaridad con los idiomas objetivo ayuda mucho a discriminarlos de entre los idiomas no objetivos.
- Cuando no se está lo suficientemente familiarizado con un idioma objetivo el conocimiento lingüístico no es de mucha utilidad.

En la siguiente sección veremos los algoritmos computacionales que tratan de sustituir al ser humano como el mejor reconocedor de la lengua.

2.4. Técnicas empleadas

El reconocimiento de idioma se nutre de la multitud de técnicas de reconocimiento de patrones que han sido desarrolladas durante la últimas décadas, especialmente las desarrolladas para reconocimiento de locutor, debido a su similitud con el reconocimiento de idioma (únicamente cambiamos los locutores por idiomas, la señal de audio y su parametrización siguen siendo las mismas). En esta sección presentamos las técnicas mas utilizadas por los diversos grupos de investigación en el área de reconocimiento o verificación de la lengua, pero con especial énfasis en las que han sido desarrolladas a lo largo del proyecto. Antes de entrar en detalles sobre las técnicas más utilizadas, las diferenciaremos en generativas (o informativas) y discriminativas, como veremos a continuación.

2.4.1. Modelos generativos frente a discriminativos

Debido a que el proyecto investiga el empleo del criterio discriminativo MMI para su uso en reconocimiento de idioma, se hace indispensable diferenciar los paradigmas de entrenamiento: discriminativos versus generativos.

Dado un conjunto de entrenamiento compuesto por numerosos vectores de características $\vec{x}$ todos correspondientes a una determinada categoría o clase (idioma, locutor, etc), un modelo generativo es un modelo puramente estadístico que intenta reproducir lo mas fielmente posible la distribución de probabilidad conjunta entre los datos o vectores de entrenamiento y la categoría a la que pertenecen, $p(\vec{x}|clase)$, típicamente a través del criterio de máxima verosimilitud (ML). Un GMM (Gaussian Mixture Model) cuyos parámetros son estimados a través del criterio ML, intentará reproducir la densidad de probabilidad de los datos a través de una suma ponderada de Gaussianas.

Puesto que la función de probabilidad conjunta entre los datos de entrenamiento y la clase a la que pertenecen es conocida, es posible "generar" nuevos datos que se ajusten a la distribución de probabilidad de los datos de entrenamiento, y de ahí que se lo denomine generativo.

Como ejemplo de modelos generativos (se destacan en cursiva las técnicas que han sido implementadas en el proyecto) se pueden citar los siguientes:

1. Modelos Ocultos de Markov (HMM's).
2. *Modelos de Mezclas de Gaussianas (GMM's) entrenadas por ML y Modelos de Mezclas de Gaussianas con UBM (Universal Background Model).*

3. Modelos de lenguaje en PRLM's.

4. Análisis discriminativo lineal (LDA).

Los modelos discriminativos en cambio, intentan predecir, que no modelar, la clase o categoría a la que pertenece el vector de características $\vec{x}$ modelando la función de distribución condicional $p(clase|\vec{x})$ a través de las fronteras de separación de las clases. Para modelar $p(clase|\vec{x})$ no se requiere conocer la verdadera forma funcional de la distribución de probabilidad conjunta entre la clase y $\vec{x}$, por lo que tampoco es posible generar nuevas muestras de vectores de características. Un SVM (Support Vector Machine) es un ejemplo de un clasificador binario lineal discriminativo que intenta encontrar el hiperplano (o frontera) que separa las dos clases. Un GMM entrenando por el criterio MMI (base de este PFC), es otro ejemplo de clasificador discriminativo en donde solo se modela la distribución de probabilidad de los vectores que diferencian las clases (esto es, lo vectores que determinan las fronteras de separación).

Algunos ejemplos de modelos discriminativos (en cursiva están marcadas las técnicas que han sido implementadas en el proyecto) son:

1. Redes Neurales Artificiales.

2. *Modelos de Mezclas de Gaussianas discriminativas (entrenadas según el criterio MMI).*

3. *Maquinas de Vectores Soporte.*

4. Regresión Logística.

2.4.2. Técnicas acústicas

Todas las técnicas que se describen a continuación únicamente hacen uso de información acústica (en la sección 2.2.2 se ofrecen mas detalles).

2.4.2.1. Técnicas acústicas generativas

En esta sección se describen las técnicas mas importantes que trabajan a nivel acústico y utilizan modelos generativos.

2.4.2.1.1. Modelos de Mezclas de Gaussianas (GMM)

Un GMM [3] es un modelo semi-paramétrico de aprendizaje supervisado (esto es, que de cada vector se conoce la clase a la que pertenece) que permite estimar la distribución de probabilidad de los datos de entrenamiento como una combinación lineal de distribuciones de probabilidad Gaussianas D-dimensionales (siendo D la dimensión del vector de características).

Matemáticamente un GMM de parámetros λ queda definido por la siguiente ecuación:

$$p(\vec{x}|\lambda) = \sum_{i=1}^{M} w_i p_i(\vec{x}), \tag{2.5}$$

donde $\vec{x}$ es un vector de características D-dimensional, $p_i(\vec{x}); i = 1, \dots M$ son las componentes de densidad de probabilidad, y w_i los pesos de las mezclas. Cada componente de densidad es una función Gaussiana D-dimensional de la forma:

$$p_i(\vec{x}) = \frac{1}{(2\pi)^{D/2}|\sum_i|^{1/2}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_i)' \sum_i^{-1}(\vec{x}-\vec{\mu}_i)}, \tag{2.6}$$

siendo $\vec{\mu}$ el vector de medias y Σ_i la matriz de covarianza. Por consiguiente, un GMM puede ser especificado colectivamente por los parámetros $\lambda = \{w_i, \vec{\mu}_i, \Sigma_i\} \quad i = 1, \ldots, M$ sujetos a las restricciones $\sum_{i=1}^{M} w_i = 1$, $0 \leq w_i \leq 1$, y $\int_{-\infty}^{\infty} p_i(\vec{x}) d\vec{x} = 1$.

La matriz de covarianza Σ_i utilizada se ha considerado diagonal en todos los experimentos. Esto se debe a que experimentalmente se ha confirmado que una matriz de covarianza diagonal produce mejores resultados que una completa. Además, con esto también reducimos considerablemente el numero de cálculos en coma flotante necesarios para el entrenamiento y evaluación del GMM (que se deben llevar a cabo durante el calculo de la inversa de la matriz de covarianza).

Para cada idioma se entrenará un modelo GMM que reproducirá la distribución de todos los vectores de características del mismo.

Supuesto que existan un total de T vectores disponibles para el entrenamiento, se realizarán generalmente 6 iteraciones por medio del algoritmo Baum-Welch, utilizando las siguientes ecuaciones:

$$w_i = \frac{1}{T}\sum_{t=1}^{T}\gamma_i(t) \quad \vec{\mu}_i = \frac{\sum_{t=1}^{T}\gamma_i(t)\vec{x}(t)}{\sum_{t=1}^{T}\gamma_i(t)} \quad \vec{\sigma}^2 = \frac{\sum_{t=1}^{T}\gamma_i(t)\vec{x}^2(t)}{\sum_{t=1}^{T}\gamma_i(t)} - \vec{\mu}_i^2 \; i = 1, \ldots M, \tag{2.7}$$

siendo $\gamma_i(t) = \frac{w_i p_i(\vec{x}(t))}{\sum_{k=1}^{M} w_k p_k(\vec{x}(t))}$ la probabilidad de ocupación de la mezcla i-ésima en el frame $\vec{x}(t)$ y M el número de mezclas.

Cuantos mas datos tengamos mas mezclas habrá que utilizar para obtener resultados precisos. Antes de aplicar estas ecuaciones los parámetros del modelo deben ser inicializados por medio de, por ejemplo, K-means (en la sección "Arquitectura de los sistemas MMI y SuperVectors" se indica el procedimiento que se siguió para inicializar los parámetros).

En las siguientes figuras se han representado los datos de entrenamiento de un total de 40 ficheros (obtenidos de la evaluación de idioma de NIST 2003) para los idiomas de español e inglés. Se representan el primer coeficiente cepstral c_1 MFCC en el eje X, frente al tercer coeficiente cepstral c_3 MFCC en el eje Y. A continuación se entrenan sendos modelos GMM. En las figuras de abajo se representan las curvas de nivel de densidad de probabilidad para los GMM's de 8 y 128 mezclas. Cuanto mas mezclas, mayor es el nivel de detalle del GMM en la estimación de la distribución de los datos de entrenamiento.

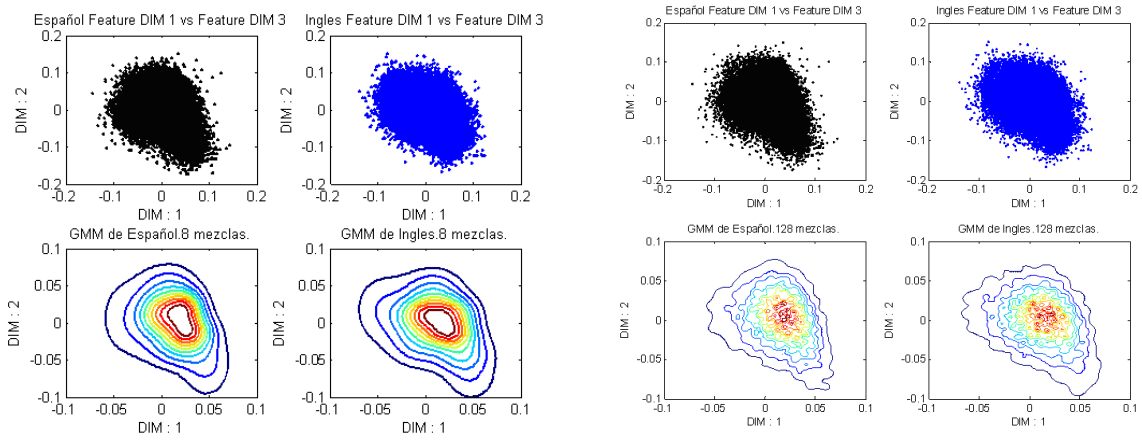

Figura 2.9: Izquierda, distribución de datos para los idiomas de español e ingles. Derecha, modelos GMM correspondientes de 8 y 128 mezclas.

Como se puede observar las distribuciones de datos para ambos idiomas son muy parecidas y están altamente solapadas. Esto hace que el reconocimiento sea complicado.

En la figura, a continuación, se muestran las regiones de clasificación de ambos modelos de idioma, obtenidas haciendo un barrido del plano $x = -0{,}1 : 0{,}1; y = -0{,}1 : 0{,}1$. El GMM cuya probabilidad sea mayor en un punto de dicho plano es el ganador (y se dibuja el punto correspondiente en azul para el inglés, y en verde para el español).

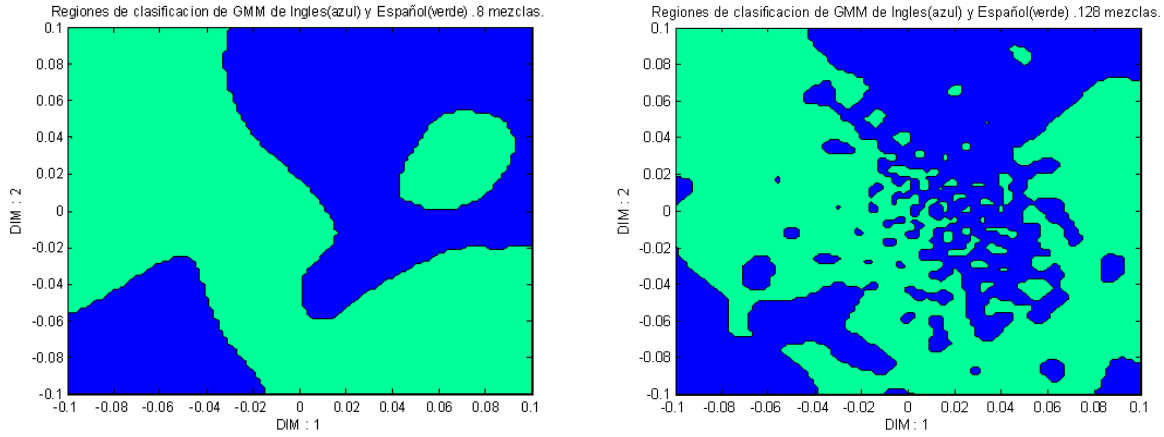

Figura 2.10: Regiones de clasificación para los GMM's de español e inglés con 8 y 128 mezclas respectivamente.

También, el aspecto en 3D (vistas desde arriba) de las funciones de densidad de probabilidad de los GMM's de español e inglés para 8 mezclas. Obsérvese como los modelos son realmente parecidos (pero aún con sutiles diferencias), debido al alto solapamiento de los datos de entrenamiento.

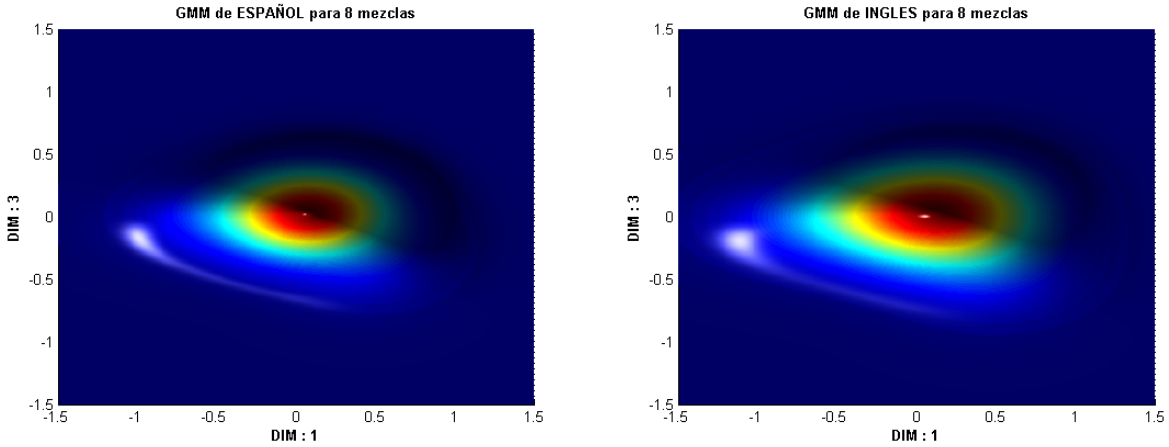

Figura 2.11: Modelos 3D (vistos desde arriba) de los GMM's de español e inglés.

2.4.2.1.2. Modelos de Mezclas de Gaussianas con UBM (GMM-UBM)

La técnica GMM-UBM fue introducida por Reynolds et Al [4] para su uso en reconocimiento de locutor, y desde entonces también se ha venido utilizando en reconocimiento de idioma (debido a las similitudes entre ambos problemas), bien directamente o indirectamente (por ejemplo en el sistema GMM-SVM, como se verá posteriormente).

La técnica GMM-UBM consiste básicamente en entrenar un modelo genérico de idioma (modelo universal) utilizando el algoritmo Baum-Welch (entrenamiento EM o ML) para posteriormente obtener para cada idioma un GMM adaptando los parámetros del UBM por medio de MAP (criterio Maximum A Posteriori) utilizando todo el audio disponible para el entrenamiento de dicho idioma.

Para obtener el modelo UBM, que representa las características comunes de todos los idiomas, se puede utilizar todo el audio disponible (de todos los idiomas) para entrenar un modelo por el criterio ML (utilizando el algoritmo Baum-Welch), o bien combinar los parámetros de los modelos GMM de idioma individuales. En el último caso lo que se hace es concatenar los parámetros de los GMM de idiomas individuales para formar nuevos parámetros de mayor dimensión. El vector de pesos se deberá dividir por el número de idiomas agregados para que su suma siga siendo unitaria.

En la siguiente figura se muestran ambos procedimientos:

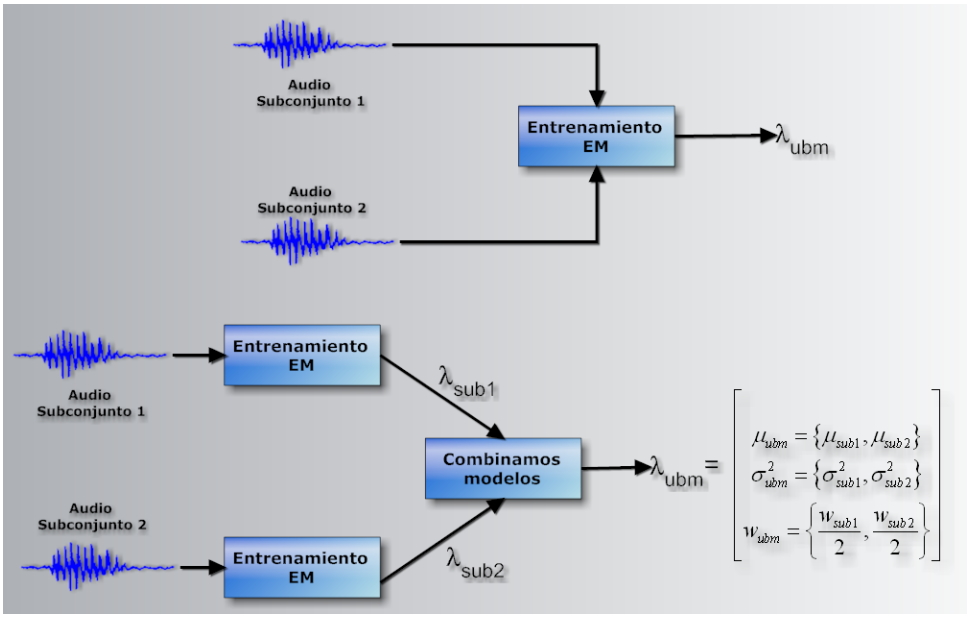

Figura 2.12: Distintas formas de obtener el UBM. La primera (imagen superior) usando el audio de todos los idiomas. La segunda (imagen inferior) concatenado los parámetros del los modelos de idioma individuales.

En los experimentos se a utilizado la primera opción (utilizar el audio de todos los idiomas para entrenar el modelo GMM Universal) por ser la mas probada por los equipos de investigación (y además más sencilla).

Una vez obtenido el UBM, para adaptar sus parámetros a un idioma en particular, utilizamos las siguientes ecuaciones, de modo análogo a la sección "Modelos de Mezclas de Gausianas (GMM)":

$$\hat{w}_i = [\frac{\alpha_i n}{T} + (1-\alpha_i)w_i]\eta \tag{2.8}$$

$$\hat{\mu}_i = \alpha_i E_i(x) + (1-\alpha_i)\mu_i \tag{2.9}$$

$$\hat{\sigma}_i^2 = \alpha_i E_i(x^2) + (1-\alpha_i)(\sigma_i^2 + \mu_i^2) - \hat{\mu}_i^2 \tag{2.10}$$

$$n_i = \sum_{t=1}^{T} \gamma_i(t) \tag{2.11}$$

$$E_i(x) = \frac{1}{n_i}\sum_{t=1}^{T} \gamma_i(t)x_t \tag{2.12}$$

$$E_i(x^2) = \frac{1}{n_i}\sum_{t=1}^{T} \gamma_i(t)x_t^2 \tag{2.13}$$

$$\gamma_i(t) = \frac{w_i p_i(\vec{x}(t))}{\sum_{k=1}^{M} w_k p_k(\vec{x}(t))} \tag{2.14}$$

siendo T el numero de vectores $x(t)$ disponibles para el entrenamiento del idioma, $\alpha_i = \frac{n_i}{n_i+r}$ el coeficiente de adaptación, y r, el "factor de relevancia" que debe ser ajustado en función de la cantidad de audio disponible para adaptar el UBM al modelo de idioma de interés. En los experimentos se ha empleado un factor $r = 14$.

En la siguiente figura, se observan las mezclas (elipses) del UBM, junto con los vectores de entrenamiento (x's) específicos de un idioma en particular, antes y después de aplicar MAP.

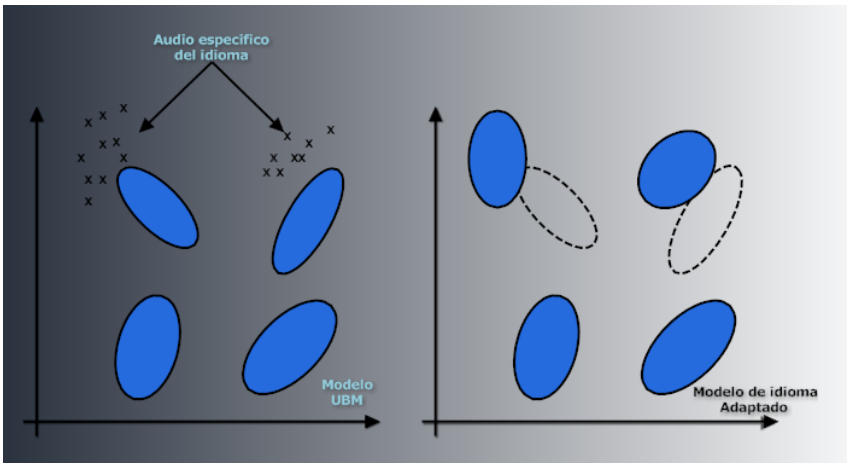

Figura 2.13: Esquema del proceso de adaptación MAP (figura adaptada de Reynolds [4]).

Como se deduce comparando ambas figuras, solo las mezclas que tengan vectores de entrenamiento cercanos o solapados, y por lo tanto un n_i no despreciable, serán modificadas considerablemente, mientras que aquellas que carezcan de vectores próximos no cambiaran ($n_i \approx 0$).

Aunque el uso de la técnica GMM-UBM, frente a utilizar GMM's independientes, ha aportado mejoras considerables en el área de reconocimiento de locutor, no esta muy claro sus beneficios en reconocimiento de idioma (los numerosos estudios que hay al respecto producen resultados dispares) cuando se comparan con modelos GMM-ML independientes. En los experimentos realizados durante el desarrollo de este PFC los GMM-UBM obtienen resultados muy parecidos a utilizar modelos independientes.

Sin embargo, el uso de un UBM para producir una representación compacta de un fichero de audio a través del vector de medias obtenido del UBM por MAP (lo que se conoce generalmente como GMM-SuperVector) combinado con técnicas mas avanzadas (como Maquinas de Vectores Soporte, análisis PCA (Principal Component Analysis) para compensación de canal, etc) si ha demostrado producir resultados sorprendentes (como se verá en los resultados experimentales). Por último, debido a la fuerte correspondencia que existe entre las mezclas del GMM del modelo UBM y el modelo de idioma adaptado de él, es posible evaluar un fichero de test utilizando únicamente las C mezclas más pesadas (con más probabilidad) del UBM para posteriormente computar únicamente las mismas mezclas en el modelo adaptado. El score o puntuación se obtiene entonces, como:

$$score_{idioma} = p^{C\_mezclas}(X|idioma) - p^{C\_mezclas}(X|UBM) \tag{2.15}$$

Supuesto que los modelos tengan M mezclas y haya N modelos, esto supone el cálculo de $M + N \cdot C$ frente a $N \cdot M$ mezclas, acelerando considerablemente el proceso de evaluación.

En la siguiente figura se ilustra el proceso de verificación usando un UBM:

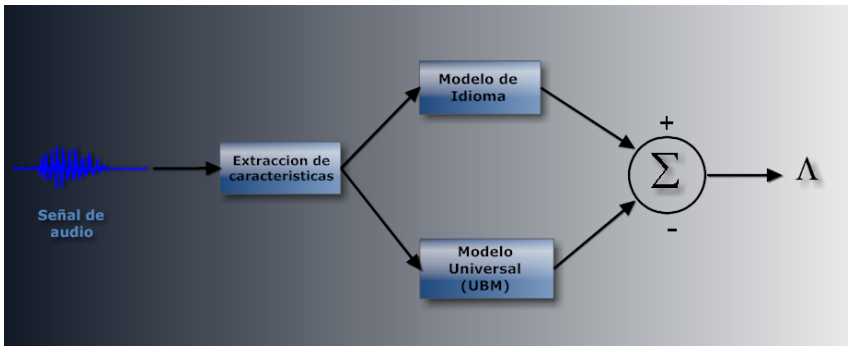

Figura 2.14: Proceso de verificación usando un UBM.

Finalmente, aprovechando los datos generados para la sección 4.3.2 del sistema discriminativo GMM-MMI, mostramos los modelos UBM y adaptados, resultantes de los mismos.

Los datos de las dos clases (véase la sección mencionada para más detalles) son:

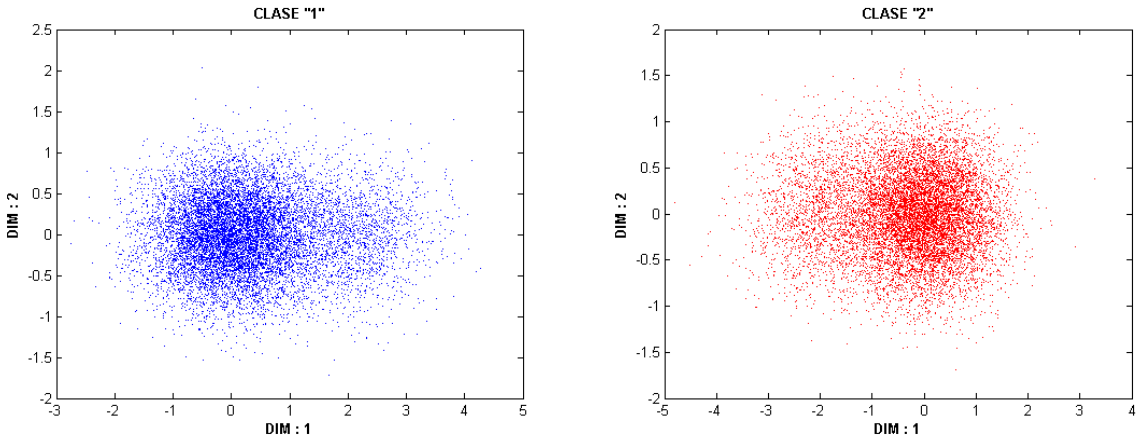

Figura 2.15: Datos de entrenamiento para las clases 1 y 2.

Y las curvas de nivel del UBM y modelos adaptados:

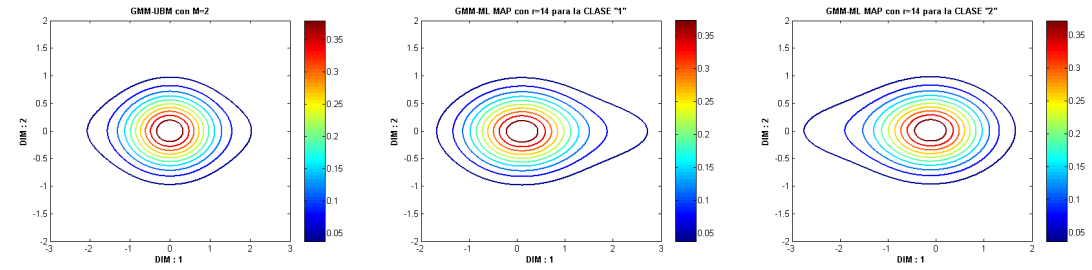

Figura 2.16: Curvas de nivel de densidad de probabilidad para el GMM del modelo UBM (izquierda) y GMM's adaptados.

Podemos observar, como era de esperar, que el UBM se ha desplazado en la dirección donde

se encontraban los vectores de datos de cada clase.

2.4.2.2. Técnicas acústicas discriminativas

En esta sección se describen las técnicas mas importantes que trabajan a nivel acústico y al mismo tiempo utilizan modelos discriminativos.

2.4.2.2.1. Modelos de Mezclas de Gaussianas Discriminativos (DGMM-MMI)

Aunque en el capitulo "El paradigma del entrenamiento MMI" se tratará en profundidad los GMM's discriminativos, en esta sección se resumen las ecuaciones matemáticas que se usaron para su implementación.

Un GMM discriminativo (DGMM), es un GMM cuyos parámetros se han entrenado según el criterio discriminativo MMI (Maximum Mutual Information), a diferencia de un GMM obtenido o bien por el criterio ML,o bien por el criterio MAP a partir de un UBM (ambos descritos en secciones anteriores).

A diferencia del criterio ML, que intenta maximizar la probabilidad entre los datos de entrenamiento de un idioma y su modelo GMM correspondiente, la función objetivo MMI intenta maximizar la probabilidad a posteriori de todos los segmentos de entrenamiento correctamente reconocidos, esto es:

$$F_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{p_\lambda(O_r|s_r)^{K_r} P(s_r)}{\sum_{\forall s} p_\lambda(O_r|s)^{K_r} P(s)} = \sum_{r=1}^{R} \log \frac{p_\lambda(O_r|s_r)^{K_r}}{\sum_{\forall s} p_\lambda(O_r|s)^{K_r}} \quad [P(s_r) = P(s)], \quad (2.16)$$

donde $p_\lambda(O_r|s_r) = \prod_{t=1}^{T_r} \sum_{j=1}^{J_s} c_{s_r j} N(o(t); \mu_{s_r j}, \sigma^2_{s_r j})$ es la probabilidad del r-ésimo segmento de entrenamiento,O_r, conteniendo el idioma s_r y parámetros del modelo $\lambda = (c_s, \mu_{s_{rj}}, \sigma^2_{s_{rj}})$. R es el número total de segmentos. El factor $0 < K_r < 1$ se determina experimentalmente para mejorar el rendimiento. En los experimentos $K_r = \frac{C}{T_r}$, con C una constante dependiente del tipo de parámetros (se uso C=6) y T_r el número de frames del segmento r-ésimo. Quitando el denominador a la función objetivo F_{MMI} obtendríamos el criterio ML, esto es:

$$f_{ML}(\lambda) = \sum_{r=1}^{R} \log P(O_r^{T_r}|s_r) \quad (2.17)$$

Mientras que en ML, se usan vectores de características individuales, en MMI se emplean segmentos, que pueden ser o bien ficheros completos (típicamente con duraciones menores o iguales a 30 segundos), o bien una porción continua de un fichero (obtenida delimitando el audio entre dos silencios, por ejemplo).

El valor de la función objetivo $F_{MMI}(\lambda)$ puede ser incrementado utilizando una versión extendida del algoritmo Baum-Welch. Las ecuaciones para actualizar las medias y varianzas son:

$$\hat{\mu}_{sm} = \frac{\theta^{num}_{sm}(O) - \theta^{den}_{sm}(O) + D_{sm}\mu'_{sm}}{\gamma^{num}_{sm} - \gamma^{den}_{sm} + D_{sm}} \quad (2.18)$$

$$\hat{\sigma}^2_{sm} = \frac{\theta^{num}_{sm}(O^2) - \theta^{den}_{sm}(O^2) + D_{sm}(\sigma'^2_{sm} + \mu'^2_{sm})}{\gamma^{num}_{sm} - \gamma^{den}_{sm} + D_{sm}} - \hat{\mu}^2_{sm}, \quad (2.19)$$

donde s y m representan la identidad del modelo (idioma) y componente de la mezcla, ${\mu'}_{sm}$ y ${\sigma'}^2_{sm}$ las antiguas medias y varianzas, y, por último, D_{sm} es una constante que controla la velocidad de convergencia, y su valor se elige como el mayor entre (1) dos veces el valor que asegure que todas las nuevas varianzas $\hat{\sigma}^2_{sm}$ sean positivas (2) $E\gamma^{den}_{sm}$, donde E es otra constante (en los experimentos se uso $E = 2$). La ecuación para los pesos del GMM se omite al comprobarse experimentalmente que estos tienen muy poco impacto en el incremento del valor de la función objetivo, y, por consiguiente, en el rendimiento. Las ecuaciones:

$$\theta^{num}_{sm}(O) = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{num}_{smr}(t)o_r(t) \tag{2.20}$$

$$\theta^{num}_{sm}(O^2) = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{num}_{smr}(t)o_r(t)^2 \tag{2.21}$$

$$\gamma^{num}_{sm} = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{num}_{smr}(t), \tag{2.22}$$

representan los estadísticos ,para las mezclas del GMM, de primer y segundo orden y la probabilidad de ocupación correspondiente al numerador de la función objetivo MMI, respectivamente. Estos estadísticos coinciden con las del criterio ML visto en las secciones anteriores. De forma similar, los estadísticos del denominador se calculan como:

$$\theta^{den}_{sm}(O) = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{den}_{smr}(t)o_r(t) \tag{2.23}$$

$$\theta^{den}_{sm}(O^2) = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{den}_{smr}(t)o_r(t)^2 \tag{2.24}$$

$$\gamma^{den}_{smr} = \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{den}_{smr}(t), \tag{2.25}$$

donde:

$$\gamma^{num}_{smr}(t) = \begin{cases} \gamma^{num}_{smr}(t) \text{ si } \mathrm{s} = \mathrm{s_r} \\ 0 \qquad\quad \text{si } s \neq s_r \end{cases} \tag{2.26}$$

$$\gamma^{den}_{smr}(t) = \gamma_{smr}(t)\frac{p_{\lambda'}(O_r|s)^{K_r}}{\sum_{q=1}^{L} p_{\lambda'}(O_r|q)^{K_r}} \tag{2.27}$$

$$\gamma_{smr}(t) = W_s\frac{c_{sm}N(o_r(t);\mu'_{sj},{\sigma'}^2_{sj})}{\sum_{j=1}^{J_s} c_{sj}N(o_r(t);{\mu'}_{sj},{\sigma'}^2_{sj})} \tag{2.28}$$

El símbolo $\lambda^{'}$ representa los parámetros antiguos y J_s el número de mezclas del modelo s.El factor $\frac{p_{\lambda'}(O_r|s)^{K_r}}{\sum_{q=1}^{L} p_{\lambda'}(O_r|q)^{K_r}}$ es el que determina la importancia de la segmentación en MMI durante el entrenamiento.

Uno de los inconvenientes de MMI, es que aprende las cantidades de audio usadas para entrenar cada modelo de idioma. Ecualizando las cantidades de audio usadas durante el entrenamiento se evitaría este problema. Sin embargo, es posible simular la misma cantidad

de audio utilizando el factor W_s en la ecuación $\gamma_{smr}(t)$ y estableciendo su valor inversamente proporcional a la cantidad de audio disponible para el idioma s.

Las diferencias más notables que hacen que los modelos entrenados por MMI obtengan mejor rendimiento se resumen en los dos puntos siguientes:

1. A diferencia de ML, con MMI los parámetros se optimizan para el correcto reconocimiento de segmentos completos (no frames individuales como en el caso de ML), teniendo en cuenta la importancia de la segmentación (palabras en su mayoría) de un idioma.

2. Solo se modelan los vectores de características que diferencian un idioma del resto y no se malgastan mezclas en vectores que comparten todas las clases. Este punto se verá con más detalle en secciones posteriores.

Por el contrario, el entrenamiento discriminativo MMI también tiene su puntos negativos, principalmente su mayor coste computacional durante el entrenamiento al involucrar todas las clases al mismo tiempo, a diferencia del criterio ML donde solo una clase esta involucrada. También tiene la tendencia de empeorar cuando se incrementan el número de clases o no se dispone del audio suficiente para la discriminación (típicamente cuando este es menor de 1 hora).

2.4.2.2.2. Maquinas de Vectores Soporte (SVM)

Las Maquinas de Vectores Soporte [22] es otra técnica de reconocimiento de patrones recientemente introducida en el campo del reconocimiento de idioma (incluso las bases matemáticas de la misma, desarrolladas principalmente por Vladimir Vapnik, son muy nuevas).

Básicamente, un SVM es un clasificador binario discriminativo construido como una combinación lineal de funciones Kernel $K(\vec{x}_i, \vec{x}_j) = \langle \Phi(\vec{x}_i), \Phi(\vec{x}_j) \rangle$ (<,> representa el producto interno), de la forma:

$$f(\vec{x}) = w^T \Phi(\vec{x}) + b = \sum_{r=1}^{R} \alpha_r y_r K(\vec{x}, \vec{x}_r) + b, \tag{2.29}$$

donde y_r representan las salidas deseadas, $\sum_{r=1}^{R} \alpha_r y_r = 0$ con $\alpha_i > 0$ y R el número total de vectores de entrenamiento. $\vec{x}_r$ son los vectores soporte y se obtienen de los datos de entrenamiento por un proceso de optimización. Las salidas deseadas son los valores +1 o -1 dependiendo de si el correspondiente vector soporte pertenece a la clase 0 o la clase 1, respectivamente. La verificación se realiza comparando el valor de la función $f(\vec{x})$ (score) con un umbral.

Más concretamente, un SVM define un hiperplano (p-1)-dimensional que permite separar los vectores de un idioma de dimensión p-dimensional (que serán los vectores de la clase 0, o usuarios) respecto del resto de idiomas (clase 1, o de impostores). Un hiperplano puede ser definido por el conjunto de puntos $\vec{x}$ que satisfagan la ecuación $\overrightarrow{w} \cdot \overrightarrow{x} - b = 0$, donde $\cdot$ representa el producto escalar de los vectores $\vec{w}$y $\vec{x}$, siendo $\vec{w}$ el vector normal al plano. El parámetro $\frac{b}{\|\vec{w}\|}$ determina el desplazamiento del vector $\vec{w}$ respecto del origen. Debido a que puede ser imposible hallar un hiperplano que separare los puntos de ambas clases en la dimensión de $\vec{x}$, se emplea la función del Kernel $\Phi(\vec{x})$ que mapea el vector $\vec{x}$ a una dimensión mayor en la que tal hiperplano existe.

En la siguiente figura se ilustra la necesidad del Kernel para encontrar el hiperplano:

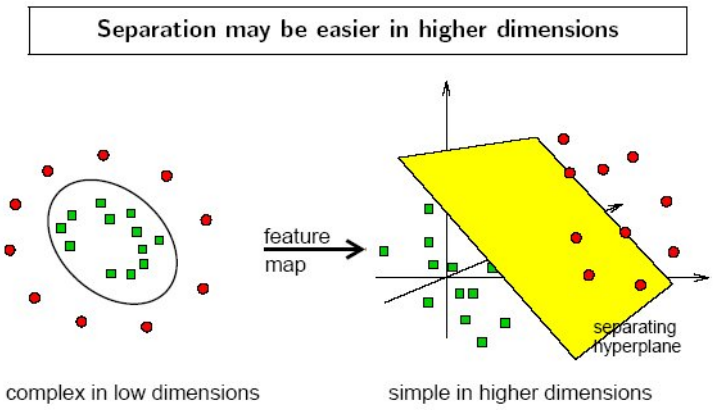

Figura 2.17: Concepto de Kernel de un SVM

El kernel $K(\vec{x}_i, \vec{x}_j)$ para ser considerado como tal debe satisfacer las condiciones de Mercer, una de ellas es que debe poder ser expresado como $K(\vec{x}_i, \vec{x}_j) = \langle \Phi(\vec{x}_i), \Phi(\vec{x}_j) \rangle = \Phi(\vec{x}_i)^T \Phi(\vec{x}_j)$.

Aunque puede haber infinitos hiperplanos que separen los dos conjuntos de puntos, sin embargo, solo estamos interesados en encontrar el hiperplano óptimo, que será el que logre la máxima separación (margen) entre las dos clases. Esto significará que el hiperplano óptimo de separación será aquel para el cual la distancia de los puntos mas próximos sea máximo. Si tal hiperplano existe, será claramente el hiperplano de máximo margen y al clasificador linear resultante de dicho criterio se le conoce como Clasificador de Margen Máximo.

En la siguiente imagen se ilustra la teoría mencionada. Los puntos negros y blancos representan cada una de las dos clases (impostores y usuarios). El hiperplano H3 no consigue separar las dos clases. El hiperplano H1 lo logra, pero por un margen muy reducido. Por ultimo, el hiperplano H3 es el de máximo margen (hiperplano óptimo).

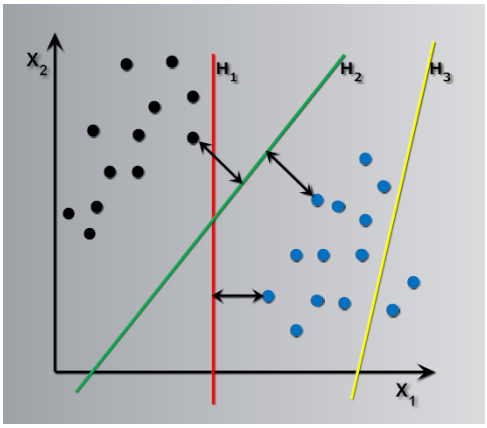

Figura 2.18: Hiperplano de máximo margen (óptimo).

A la luz de la naturaleza geométrica de un SVM, se entiende por que los vectores $\vec{x}$ que cumplen $\vec{w} \cdot \vec{x} - b = 1$ o bien $\vec{w} \cdot \vec{x} - b = -1$ (donde +1 y -1 son las etiquetas para las clases 0 y 1) se les denomina vectores soporte: estos son los vectores que se encuentran mas próximos al hiperplano y sobre los que este descansa (esto es, "soportan el peso" del hiperplano).

En la siguiente figura se muestran tales vectores:

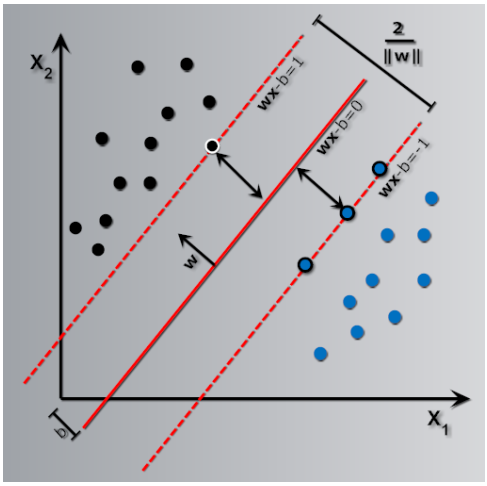

Figura 2.19: Vectores Soporte.

2.4.2.2.3. GMM-SVM (SuperVector-SVM)

Para poder aplicar el clasificador binario SVM a reconocimiento de idioma, debemos ser capaces de obtener una representación compacta de los parámetros de un fichero de audio. Esto es así porque un fichero de audio puede tener una duración variable, y uno de los requerimientos de un SVM es que los vectores deben tener la misma dimensión. Una forma de solucionar esto es por medio de lo que se conoce como SuperVector, y que ha dado muy buenos resultados.

Básicamente, un GMM-SVM consiste en entrenar para cada idioma un SVM utilizando Supervectores como datos de entrenamiento. Los Supervectores, son obtenidos por el siguiente procedimiento:

1. Se entrena un modelo universal (UBM) de M mezclas utilizando todo el audio disponible.

2. Por cada fichero de parámetros se obtiene un GMM de M mezclas por medio de MAP. Solo las medias son adaptadas del UBM.

3. De cada GMM adaptado se forma un SuperVector concatenando el vector de medias normalizado por la desviación típica correspondiente, esto es:

$$\vec{SuperVector} = [\frac{\mu_1}{\sigma_1}, \frac{\mu_2}{\sigma_2}, \cdots, \frac{\mu_i}{\sigma_i}, \cdots, \frac{\mu_M}{\sigma_M}], \tag{2.30}$$

donde $\frac{\mu_i}{\sigma_i}$ representa el cociente componente a componente del vector de medias y desviación típica de la mezcla i-ésima. Si el vector de parámetros tiene dimensión D, entonces el SuperVector tendrá dimensión $M \cdot D$.

En las siguientes figuras se ilustra todo el proceso:

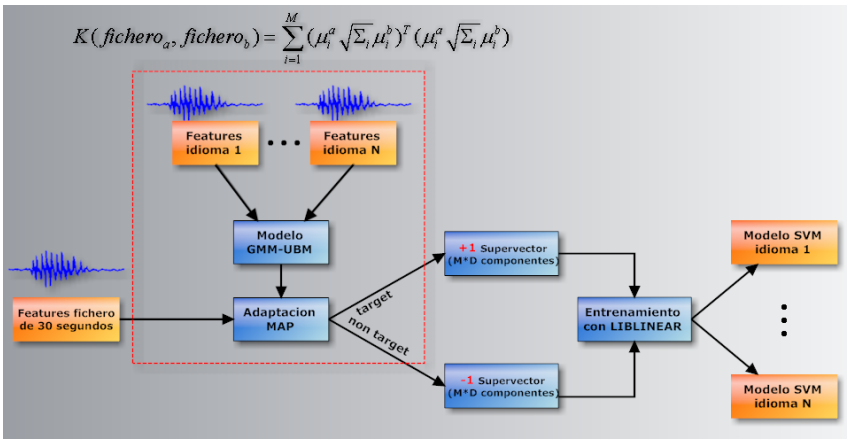

Figura 2.20: GMM (SuperVector)-SVM.

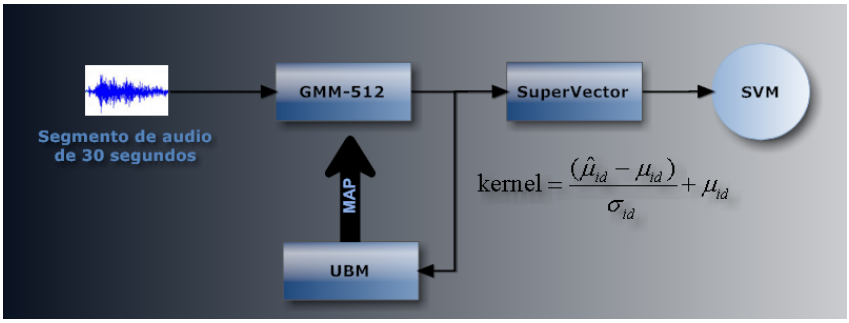

Figura 2.21: Modificación del sistema GMM-SVM. Sistema AGMM-SVM utilizado en la evaluación de Albayzin 2008.

2.4.3. Técnicas fonéticas

Los sistemas que trabajan al nivel fonético de la señal de voz, han constituido, salvo contadas excepciones, el estado del arte en reconocimiento idiomático desde su inicio. En particular, los dos primeros sistemas (con diversas variaciones) que se describen a continuación son los que han obtenido (en media) los mejores resultados durante las últimas evaluaciones NIST de idioma. Él empleo de lattices para el modelado de lenguaje o el uso de técnicas clasificativas mas potentes como las Maquinas de Vectores Soporte en combinación con el esquema básico de un PPRLM (Parallel Phone Recognition Language Modeling) que describiremos a continuación, han hecho que el rendimiento de los sistemas fonéticos y la variedad de los mismos aumente de forma considerable durante los últimos años, como atestiguan las últimas evaluaciones NIST.

A continuación se describen los sistemas fonéticos clásicos empleados en reconocimiento de idioma.

2.4.3.1. PRLM

El sistema **PRLM (Phone Recognition Language Modeling)** fue introducido por primera vez en reconocimiento de idioma por Hazen y Zue en 1993 [7]. Un año después y de forma independiente, Zissman [1][2] desarrollo un sistema similar al primero. Al igual que todos los sistemas de reconocimiento de idioma, un PRLM se compone de dos partes principales,

la primera de ellas (*front-end*) se encarga de la extracción de características relativas al idioma contenidas en la señal de voz dentro del nivel de información en el que opere dicho sistema y una segunda parte (*back-end*) que se encarga de modelar de alguna forma dichas características y de realizar la función de reconocimiento propiamente dicha.

En un PRLM el *front-end* lo constituye un reconocedor fonético que se encarga de segmentar la señal de voz en sus unidades fonéticas. Lo que hace especialmente atractivo a un PRLM para reconocimiento de idioma y a los sistemas basados en él, es que el idioma en el que está entrenado el reconocedor fonético no tiene por que coincidir con el idioma objetivo a identificar. De está forma el PRLM se puede generalizar a cualquier idioma.

Aunque existen diversas formar de construir el reconocedor fonético, lo más habitual es que se emplee para modelar los fonemas un HMM continuo de 3 estados izquierda-derecha sin bucles. Cada estado del HMM a su vez se modela con un GMM típicamente de 8 a 30 mezclas (dependiente del audio disponible). Para entrenar el reconocedor fonético es necesario disponer de suficiente audio con sus correspondientes transcripciones fonéticas (audio etiquetado) y esto ha de hacerse de forma manual por lo que puede resultar difícil y tedioso.

El siguiente componente de un PRLM, lo que se conoce como back-end, lo constituyen un conjunto de modelos de lenguaje (LM), uno para cada uno de los idiomas a identificar. Un modelo de lenguaje intentará predecir, con mas o menos atino, con qué probabilidad ocurrirá una determinada secuencia de símbolos (en nuestro caso los símbolos serán los fonemas obtenidos por el front-end del sistema PRLM) en función de cada símbolo y los que le preceden. Según esto, dada una secuencia de fonemas $F = \{f_1, f_2, ..., f_T\}$.la probabilidad de ocurrencia de la misma se expresará como:

$$p(F) = p(f_1, ..., f_T) = \prod_{i=1}^{T} p(f_i | f_1, ..., f_{i-1}), \tag{2.31}$$

siendo $p(f_i|f_1, ..., f_{i-1})$ la probabilidad de que el fonema f_i sea precedido por los fonemas $f_1, ..., f_{i-1}$ de la secuencia.

Debido a que la probabilidad $p(F)$ en la práctica resulta imposible de calcular, a menudo se recurre a aproximaciones basadas en la siguientes asunciones, a saber:

1. Que cada fonema de la secuencia no depende de ninguno de los anteriores, en cuyo caso tendremos que :

$$p(F) = p(f_1, f_2, ..., f_T) = \prod_{i=1}^{T} p(f_i) \tag{2.32}$$

2. Que cada fonema de la secuencia solo depende del que le precede, esto es:

$$p(F) = p(f_1) \prod_{i=2}^{T} p(f_i | f_{i-1}) \tag{2.33}$$

3. Que solo dependa de los dos anteriores :

$$p(F) = p(f_1)p(f_2) \prod_{i=3}^{T} p(f_i | f_{i-1}, f_{i-2}) \tag{2.34}$$

y en general, para el caso en que solo dependa de los n anteriores :

$$p(F) = p(f_1)p(f_2)...p(f_n) \prod_{i=n+1}^{T} p(f_i|f_{i-1}, f_{i-2}, ..., f_{i-n}) \tag{2.35}$$

Los casos anteriores se denominan Modelos de Lenguaje basados en unigramas, bigramas, trigramas y en general n-gramas. Un n-grama es una subsecuencia de n símbolos, y en total habrá V^n de ellos, siendo V el número de símbolos distintos. En nuestro caso V será el número total de fonemas de nuestro reconocer fonético.

Dado que ninguno de los casos anteriores es cierto, si no solo parcialmente, la probabilidad de la secuencia $F = \{f_1, f_2, ..., f_T\}$ se aproximará utilizando una combinación lineal de los casos anteriores (unigramas, bigramas y trigramas). Esto es lo que se conoce como modelo de lenguaje interpolado, siendo los dos casos siguientes los más utilizados:

1. LM interpolado basado en bigramas :

$$\hat{p}(f_t|f_{t-1}) = \alpha_2 p(f_t|f_{t-1}) + \alpha_1 p(f_t) + \frac{\alpha_0}{V} \tag{2.36}$$

2. LM interpolado basado en trigramas :

$$\hat{p}(f_t|f_{t-2}, f_{t-1}) = \alpha_3 p(f_t|f_{t-2}, f_{t-1}) + \alpha_2 p(f_t|f_{t-1}) + \alpha_1 p(f_t) + \frac{\alpha_0}{V} \tag{2.37}$$

Los coeficientes α se determinan experimentalmente y cumplen $\sum_{i=1}^{n} \alpha_i = 1$

El resto de términos se corresponden con las probabilidades de los unigramas, bigramas y trigramas y se calculan contabilizando su ocurrencia en las trascripciones obtenidas por el reconocedor fonético para cada idioma a reconocer, siendo estas:

$$p(f_t|f_{t-2}, f_{t-1}) = \frac{C(f_{t-2}, f_{t-1}, f_t)}{C(f_{t-2}, f_{t-1})} \tag{2.38}$$

$$p(f_t|f_{t-1}) = \frac{C(f_{t-1}, f_t)}{C(f_{t-1})} \tag{2.39}$$

$$p(f_t) = \frac{C(f_t)}{\sum_{i=1}^{V} C(f_i)} \tag{2.40}$$

donde $C(f_{t-2}, f_{t-1}, f_t)$ es el número de veces que aparece la secuencia la subsecuencia f_{t-2}, f_{t-1}, f_t. Del mismo modo $C(f_{t-1}, f_t)$es el número de veces que el fonema f_{t-1} es seguido por el fonema f_t. Finalmente, $C(f_t)$ es el número de veces que el fonema f_t aparece en las transcripciones de todo el audio de entrenamiento del idioma a reconocer.

Resumiendo, los pasos a seguir para obtener los modelos de lenguaje son los siguientes:

1. Todo el audio disponible para cada uno de los idiomas objetivos I_i $(i = 1, ..., L)$ a reconocer se transcribe usando el reconocedor fonético.

2. Se calcula el número de veces que cada uno de los V unigramas, V^2 bigramas y V^3 trigamas aparecen en el audio transcrito. Con estos datos se construyen los modelos de lenguaje intepolados (bigramas o trigramas) uno para cada idioma a reconocer $LM^{I_i}_{n-grama} \quad i = 1, ..., L$.Algunas secuencias de fonemas serán mas frecuentes en determinados idiomas

dando lugar a modelos de lenguaje diferentes para cada uno. Este es básicamente el principio en el que se basan la mayoría de técnicas que trabajan al nivel fonológico de la señal de voz.

Una vez obtenidos los modelos de lenguaje, para reconocer a que idioma pertenece un fichero de audio, procedemos en dos pasos:

1. Utilizando el reconocedor fonético obtenemos una secuencia de fonemas $F = \{f_1, f_2, ..., f_T\}$.
2. La secuencia de fonemas anterior es evaluada por todos los modelos de lenguaje interpolados, obteniendo las siguientes puntuaciones (scores) :
   $S(F|LM^{I_i}_{bigrama}) = \sum_{t=1}^{T} \log \hat{p}(f_t|f_{t-1}, LM^{I_i}_{bigrama})\ i = 1, ..., L$ para LM basados en bigramas.
   $S(F|LM^{I_i}_{trigrama}) = \sum_{t=1}^{T} \log \hat{p}(f_t|f_{t-2}f_{t-1}, LM^{I_i}_{trigrama})\ i = 1, ..., L$para LM basados en trigramas.

El idioma reconocido será aquel cuya puntuación (score) sea máxima.

En la siguiente figura, se resume el esquema de funcionamiento básico de un sistema PRLM.

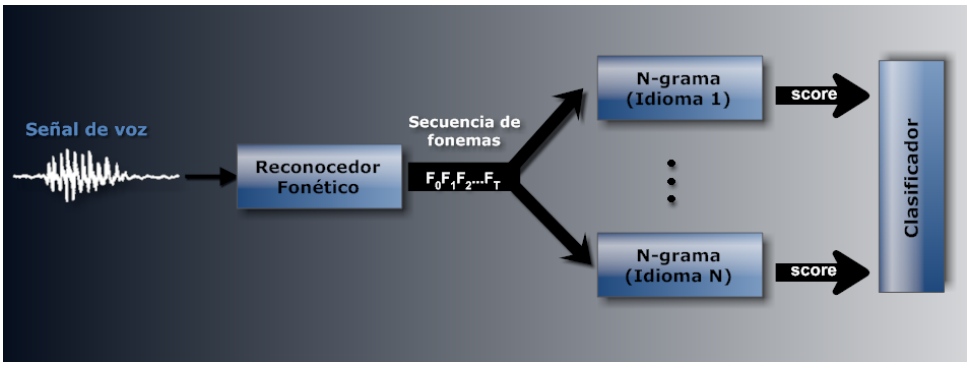

Figura 2.22: Esquema de funcionamiento de un PRLM.

Es posible sustituir el reconocedor fonético por un GMM [16][17], en donde lo que se modela son los indices de la mezcla del GMM mas probable. Se obtendrá el índice de la mezcla mas probable por cada vector de características (frame), y la secuencia de indices se utilizara para entrenar modelos de lenguaje al igual que se hace en un PRLM.A este sistema se lo conoce como GMM-Tokenizer, y tiene la ventaja frente a un PRLM de no necesitar audio transcrito para el entrenamiento.

2.4.3.2. PPRLM

Cuando se dispone del suficiente audio etiquetado para más de un idioma, es posible entrenar más de un reconocedor fonético y por consiguiente, disponer de varios PRLM. Zissman pensó que combinando varios PRLM en paralelo, lo que se conoce como PPRLM (**Parallel PRLM**) el redimiendo sería superior a cualquiera de los PRLM individuales. En efecto, es posible que un PRLM con un front-end (reconocedor fonético) entrenado con un idioma determinado funcione mejor para ciertos idiomas objetivos y peor para otros, y que para otro PRLM entrenando con un idioma diferente ocurra justo el efecto contrario (de esta forma se complementan). Se

debe hacer hincapié en el hecho de que al igual que ocurría en un PRLM, la virtud de un PPRLM radica en que no se precisa que ninguno de los idiomas objetivo tenga que coincidir con los idiomas con los que fueron entrenados los reconocedores fonéticos de los PRLM's, lo que permite en teoría aplicar esta técnica a un número ilimitado de idiomas.

La forma más común, y al mismo tiempo la mas sencilla, de combinar varios PRLM obtenidos de forma independiente, consiste en sumar los scores de los mismos idioma con idioma (esto es lo que se conoce como fusión suma).Antes de poder sumar los scores es necesario aplicar algún tipo de normalización como por ejemplo T-Norm. Después de sumar los scores se puede volver a aplicar una segunda T-normalización.

Debido a la gran carga computacional que supone tener varios reconocedores fonéticos en paralelo, en la práctica los sistemas PPRLM se ven seriamente limitados en aplicaciones en tiempo real, haciéndose necesario un compromiso entre precisión (tanto mayor cuantos mas PRLM's combinemos, aunque no de forma proporcional a su número) y tiempo computacional.

La siguiente figura ilustra el esquema básico de funcionamiento [5][6] de un PPRLM.

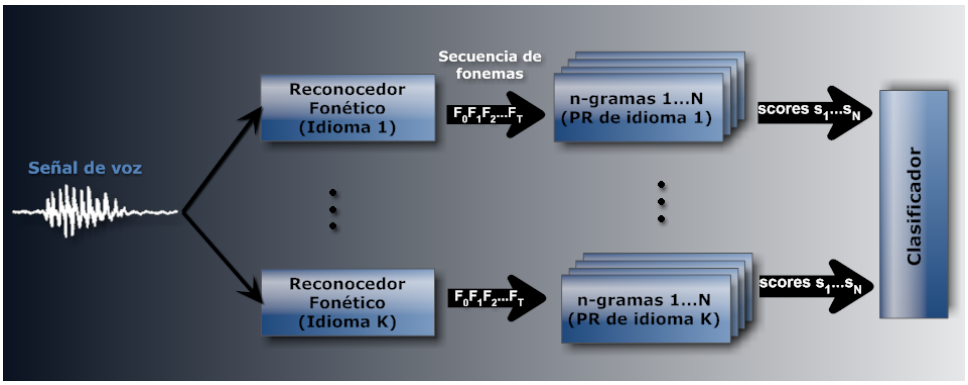

Figura 2.23: Esquema de funcionamiento de un PPRLM.

2.4.3.3. PPR

El sistema PPR (Parallel Phone Recognition) se deriva del hecho contrastado experimentalmente de que un PRLM funciona mejor reconociendo el idioma objetivo al que transcribe. De este modo en un sistema PPR todos los idiomas objetivos a reconocer tienen sus correspondientes reconocedores fonéticos entrenados para esos mismos idiomas.

Básicamente un sistema PPR consta de los mismos elementos que un PPRLM, a saber, varios reconocedores fonéticos y modelos de lenguaje en paralelo basados en n-gramas. Sin embargo, a diferencia de un PPRLM donde los n-gramas se construyen a través del audio trascrito por los diferentes reconocedores fonéticos, el disponer de audio etiquetado para todos los idiomas objetivos posibilita que estos sean entrenados de forma independiente. Esto permite que los n-gramas puedan ser integrados directamente en los reconocedores fonéticos. Los estadísticos de los n-gramas son empleados por los reconocedores fonéticos para obtener los scores finales en la fase de evaluación.

A pesar de que el rendimiento de un sistema PPR es notoriamente superior al de un PPRLM, en la actualidad los sistemas PPR son raramente utilizados en la práctica por la necesidad de tener que disponer de audio etiquetado fonéticamente para todos los idiomas a reconocer, lo que en muchas ocasiones puede resultar imposible.

2.4.3.4. Phones-SVM

El sistema Phones-SVM combina los modelos de n-gramas empleados en los sistemas PPRLM con la potencia de clasificación de los SVM (Maquinas de Vectores Soporte). A diferencia de un PPRLM, donde las probabilidades de los n-gramas se combinan para formar un modelo de lenguaje interpolado para cada idioma a reconocer, en un sistema Phones-SVM estas probabilidades son vectorizadas para entrenar un SVM, mejorándose de esta forma la precisión respecto de un PPRLM.

Al igual que un PPRLM no es más que la fusión de varios sistemas PRLM que emplean diferentes reconocedores fonéticos, el sistema Phones-SVM es en realidad la fusión de varios sistemas Phone-SVM en paralelo (siempre que dispongamos de varios reconocedores fonéticos).

Para cada idioma objetivo se obtendrá un modelo SVM, siguiendo los siguientes pasos:

1. Cada fichero de audio se pasará por el reconocedor fonético, obteniéndose la secuencia de fonemas $f_1, f_2, \ldots, f_n$. De cada fichero se obtendrá un vector que servirá para entrenar el SVM.

2. De la secuencia anterior obtenemos sus bigramas que serán $f_1f_2, f_2f_3, \ldots, f_{n-1}, f_n$ (los unigramas serán la secuencia misma). Una vez que se han obtenido los n-gramas de la secuencia, calcularemos las probabilidades de todos los n-gramas posibles que, suponiendo que el reconocedor fonético tenga M fonemas distintos, serán $d_1, d_2, \ldots, d_M$ para los unigramas, $d_1d_1, \ldots, d_Md_M$ para los bigramas, etc. Para ello aplicaremos las siguientes expresiones (solo se incluyen hasta los bigramas):

$$p(d_i) = \frac{\sum_{k=1}^{n} \delta(f_k = d_i)}{\sum_{k=1}^{n} \sum_{i=1}^{M} \delta(f_k = d_i)} \tag{2.41}$$

$$p(d_id_j) = \frac{\sum_{k=1}^{n-1} \delta(f_kf_{k+1} = d_id_j)}{\sum_{k=1}^{n-1} \sum_{k=1}^{M} \sum_{l=1}^{M} \delta(f_kf_{k+1} = d_kd_l)} \tag{2.42}$$

   donde la función $\delta(x)$ devuelve el valor 1 cuando lo que encierra entre sus paréntesis es cierto y 0 en caso contrario.

3. Las probabilidades anteriores se agrupan en un vector obteniendo :

$$\vec{v} = [p(d_1) \quad \ldots \quad p(d_M) \quad p(d_1d_1) \quad \ldots \quad p(d_Md_M)]^t$$

El vector anterior se normaliza de forma que los vectores de dos secuencias fonéticas de dos ficheros distintos sean comparables por el SVM, resultando finalmente el vector:

$$\vec{v}_{TFLLR} = \begin{bmatrix} \frac{p(d_1)}{\sqrt{p(d_1|background)}} \\ \vdots \\ \frac{p(d_M)}{\sqrt{p(d_M|background)}} \\ \frac{p(d_1d_1)}{\sqrt{p(d_1d_1|background)}} \\ \vdots \\ \frac{p(d_Md_M)}{\sqrt{p(d_Md_M|background)}} \end{bmatrix} \tag{2.43}$$

donde $p(d_1|background)$ representa la probabilidad de que el unigrama d_1 aparezca en las transcripciones de todos los ficheros de todos los idiomas y se calcula igual como si fuera para un solo fichero empleando la expresión del paso 2. De forma análoga ocurre para el resto de términos del vector $\vec{v}_{TFLLR}$.

Dicho vector se agregará al fichero de entrenamiento del modelo SVM, bien con etiqueta +1 en caso de que el fichero pertenezca al idioma del modelo o bien con etiqueta -1 en caso contrario. El mapeo del fichero de audio (que puede ser de longitud variable) al vector final (de longitud fija necesaria por definición para poder ser usada como entrada para un SVM) en la forma final de la expresión anterior se denomina Kernel TFLLR (Term Frequency Log Likelihood Ratio). Para el entrenamiento del SVM puede usarse el software LIBSVM empleando un kernel lineal.

3

Extracción de características en los sistemas de reconocimiento de idioma

En este capitulo describiremos la primera de la tres partes fundamentales de que se compone un reconocedor de patrones (y esto incluye a un reconocedor de idioma), que son la extracción de características del patrón a reconocer (en nuestro caso el idioma contenido en la señal de voz), y también conocida en la bibliografía como parametrización, a continuación el entrenamiento de modelos, y, por ultimo, la obtención de puntuaciones (evaluación).

Los parámetros MFCC que se han utilizado durante todo el proyecto, y que describiremos a continuación, son un tipo particular de coeficientes de cepstrales, todos ellos derivados de la aplicación del Cepstrum sobre una ventana de tiempo de la señal de voz (típicamente de 20 ms). Desde el punto de vista puramente matemático el cepstrum puede ser considerado un operador matemático que transforma una convolución en el dominio temporal en una suma en el dominio espectral (esto es lo que se conoce como transformación homomórfica). De esta forma se consigue separar de forma muy elegante las dos componentes de información de la señal de voz: la excitación y la respuesta en frecuencia del tracto vocal. La excitación es producida por la vibración de las cuerdas vocales en los sonidos tonales o sonoros y determinará la mayor parte de la información prosódica de la señal de voz: intensidad y tono (frecuenta fundamental). Por su parte, la respuesta en frecuencia se corresponde con una determinada configuración del tracto vocal, y, que a su vez determinará las frecuencias de resonancia (formantes) de la misma, características para cada fonema y sonido de un idioma.

La excitación en si misma se puede modelar mediante una señal con un periodo y amplitud (intensidad) determinadas, para los sonidos tonales o sonoros (en los que existe vibración de las cuerdas vocales), y mediante una fuente de ruido aleatorio para los sonidos no sonoros o sordos. Los reconocedores de idioma que trabajan a nivel de la prosodia de la señal de voz utilizan en particular la frecuencia fundamental de la excitación para caracterizar, por ejemplo, la entonación característica de ciertos idiomas.

A su vez, la respuesta en frecuencia se modela mediante un filtro. Los parámetros de este filtro son los que determinan el contenido fonético y lingüístico de la señal de voz que desea sintetizar o clasificar.

En la siguiente figura se muestra un modelo matemático simplificado del mecanismo de producción de voz. En la figura $G(z)$ simula la excitación glotal como un tren de impulsos de

amplitud A_v para los sonidos tonales y como una fuente de ruido aleatorio de amplitud A_n para los sonidos sordos. El conmutador (z) permite elegir entre ambos tipos de excitación. A continuación la señal es pasada por un filtro $Z_L(z)$ cuyos parámetros (obtenidos, por ejemplo, a través de codificación lineal predictiva o PLC) se eligen de acuerdo al fonema o sonido del discurso que se desea producir.

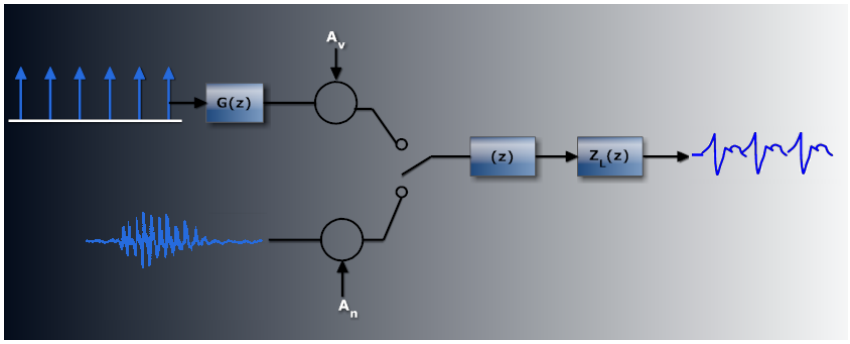

Figura 3.1: Aproximación matemática para la síntesis de voz.

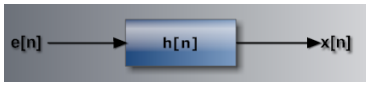

Figura 3.2: Componentes de información de la señal de voz $x[n]$: excitación $e[n]$ y filtro $h[n]$.

En general, el cepstrum se define como la transformada de Fourier inversa del logaritmo del espectro de la señal de voz, esto es:

$$Cepstrum(x[n]) = \hat{x}[n] = F^{-1}\left[\log\left(\left|F\left[x[n]\right]\right|\right)\right] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log\left|X\left(e^{j\omega}\right)\right| e^{j\omega} d\omega \tag{3.1}$$

Se puede demostrar que las dos componentes de información (de las que extraeremos las características necesarias para el reconocimiento de idioma) de la señal de voz y relacionadas por la siguiente convolución:

$$x[n] = e[n] * h[n], \tag{3.2}$$

se separan linealmente en una suma al serles aplicados el operador cepstral :

$$\hat{x}[n] = \hat{e}[n] + \hat{h}[n] \tag{3.3}$$

Aunque la mayor parte de las técnicas paramétricas de extracción de características de la señal de voz hacen uso del cepstrum, este raramente se utiliza directamente. En cambio, la mayoría de parametrizaciones realizan sobre el mismo alguna transformación con el objeto de eliminar posibles efectos del canal o emular fenómenos psicoacústicos. Esto último es lo que se lleva a cabo para el cálculo de los parámetros cepstrales MFCC, en donde se emplea una nueva escala de frecuencias denominada MEL para imitar el comportamiento psicoacústico a tonos puros de distinta frecuencia dentro del oído humano.

3.1. MFCC (Mel-Frequency Cepstral Coeficients)

Los parámetros cepstrales MFCC, se derivan directamente de la aplicación del cepstrum sobre un frame (ventana) de la señal de audio. Sin embargo, a diferencia del cepstrum, se realiza un escalado no lineal sobre el eje de frecuencias (escala de frecuencias MEL) con el objeto de imitar el comportamiento del sistema auditivo humano. Estudios psicoacusticos han demostrado que el sistema auditivo humano procesa la señal de voz en el dominio espectral, caracterizándose este por tener mayores resoluciones en bajas frecuencias y esto es precisamente lo que se consigue con el uso de la escala MEL, asignar mayor relevancia a las bajas frecuencias de forma análoga a como se hace en el sistema auditivo humano, en concreto, en el oído interno.

Davis y Mermelstein [10] fueron los que demostraron que los parámetros MFCC resultan ventajosos para el reconocimiento del habla y por extensión y afinidad, también para el reconocimiento de idioma.

El cálculo de los parámetros MFCC es relativamente sencillo. Salvo ligeras variaciones en la implementación, se pueden obtener siguiendo los siguientes pasos:

- La señal de audio a parametrizar se pasa por un filtro de pre-énfasis. El objetivo es compensar la atenuación de aproximadamente 20 dB/década que se produce en el proceso fisiológico del mecanismo de producción del habla. Este paso es opcional pero altamente recomendable para enfatizar los formantes (frecuencias de resonancia de la cavidad acústica del tracto vocal) de alta frecuencia. En las siguientes figuras observamos el espectro de una señal de voz, con y sin pre-énfasis, además del filtro en si mismo.

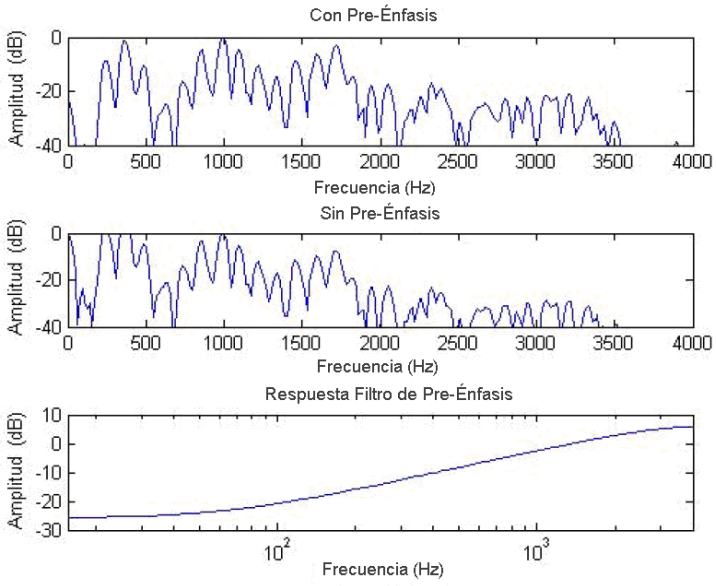

Figura 3.3: Respuesta en frecuencia del filtro de pre-énfasis.

- La señal de voz se multiplica por una ventana de hamming típicamente de 20 ms, con un desplazamiento entre ventanas de entre 10 ms. De esta forma se obtendrá un vector MFCC cada 10 ms.

La ventana de hamming tiene la forma:

$$w[n] = 0{,}54 - 0{,}46\cos(\frac{2\pi n}{N-1}) \tag{3.4}$$

Aunque se podría usar una simple ventana rectangular, se prefiere esta ventana para disminuir (que no evitar) los efectos de borde y lóbulos laterales de esta primera al calcular la DFT (transformada discreta de Fourier).

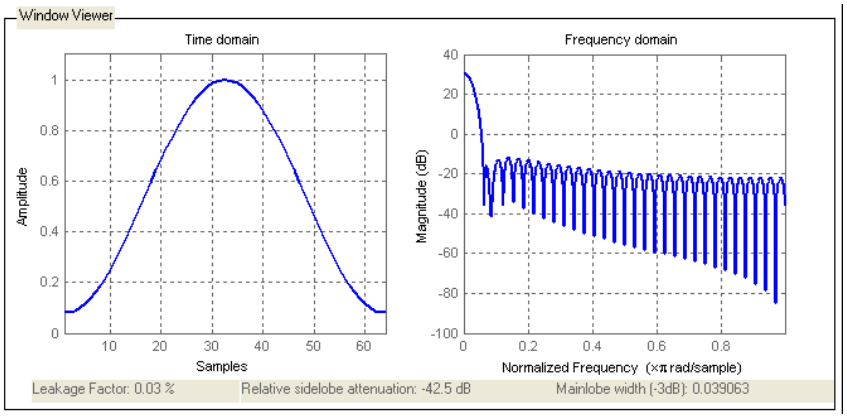

Figura 3.4: Transformada de Fourier de una ventana de Hamming.

- Calculamos la DFT de tamaño N de la señal enventanada aplicando:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi nk/N}, 0 \leq k < N \tag{3.5}$$

El número de puntos de la DTF elegido para los experimentos fue N=512.

Una vez calculada la DFT obtenemos la potencia espectral utilizando la siguiente expresión:

$$|X[k]|^2 = S[k] = (real(X[k]))^2 + (imag(X[k]))^2 \tag{3.6}$$

- A continuación la potencia espectral $S[k]$ se multiplica por un banco de Mfiltros triangulares de área unidad, espaciados de acuerdo a la escala de frecuencias MEL. La escala de frecuencias MEL como ya se dijo anteriormente imita la respuesta en frecuencia del sistema auditivo humano.

Matemáticamente, los filtros MEL se corresponden con las siguientes formulas:

$$M_l[k] = \begin{cases} 0 & k < f[l-1] \\ \frac{2(k-f[l-1])}{(f[l+1]-f[l-1])(f[l]-f[l-1])} & f[l-1] \leq k \leq f[l] \\ \frac{2(k-f[l-1])}{(f[l+1]-f[l-1])(f[l]-f[l-1])} & f[l] \leq k \leq f[l+1] \\ 0 & k > f[l+1] \end{cases}, \tag{3.7}$$

donde:

$$f[m] = \left(\frac{N}{F_s}\right) B^{-1}(B(f_1) + m\frac{B(f_h) - B(f_1)}{M+1}) \tag{3.8}$$

$$B^{-1}(b) = 700(e^{b/2595} - 1) \tag{3.9}$$

(3.10)

La función $f[m]$ determina los extremos de cada filtro triangular, siendo f_1 y $f_h$los extremos inferior y superior como se observa en la siguiente imagen:

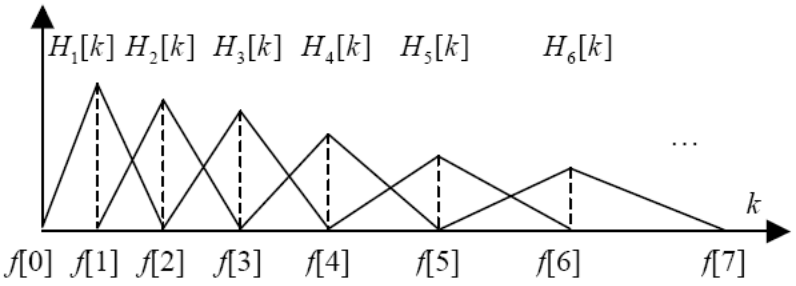

Figura 3.5: Banco de filtros MEL

Si conocemos, por ejemplo, que el audio es telefónico podremos poner f_1=300 Hz y f_h=3400 Hz, ya que sabemos que fuera de estas frecuencias no habrá información útil, salvo el ruido introducido por el canal y otras señales adversas.

La función inversa $B^{-1}(b)$ determina la anchura de los filtros de acuerdo a la escala MEL.

El comportamiento del sistema psico-acustico humano se aproxima mediante la función $mel(f)$:

$$mel(f) = 2595 \log_{10}(1 + \frac{f}{700}), \tag{3.11}$$

cuya representación grafica se corresponden con las siguientes figuras :

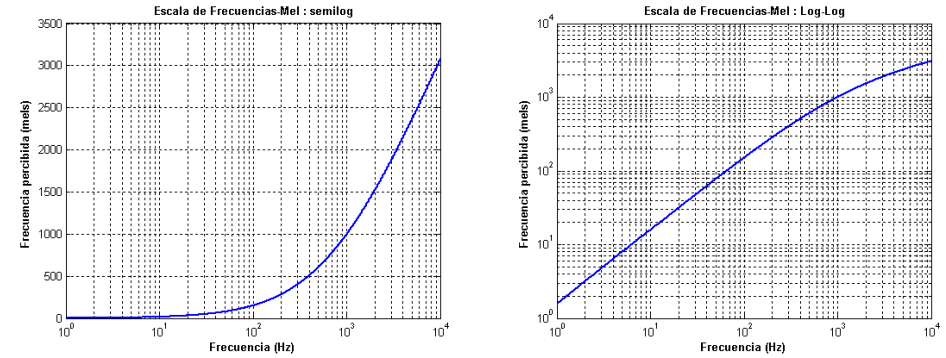

Figura 3.6: Escala de frecuencias MEL en semilog y log-log.

- Posteriormente se obtiene la energía en cada uno de los Mcanales ocupados por los filtros MEL sumando el productorio del apartado anterior. Al obtener la energía conseguimos que los parámetros MFCC calculados de este modo sean mas robustos frente a fuentes de distorsión del canal a diferencia de otros tipos de extracción de características de la señal de voz como por ejemplo los LPC (Linear Predictive Coding).

La energía en cada banda se obtiene aplicando la siguiente expresión:

$$\tilde{S}\,[l] = \sum_{k=0}^{N/2} S[k] M_l[k] \qquad l = 0, 1, \ldots, L-1 \tag{3.12}$$

1. Finalmente, de forma análoga al Ceptrum, los coeficientes MFCC se obtienen aplicando la transformada inversa del coseno (DTC-II) al logaritmo de los coeficientes de energía obtenidos en el punto anterior. Esto es :

$$c[i] = \sqrt{\frac{2}{L}} \sum_{m=1}^{L} \log(\tilde{S}[m]) \cos(\frac{\pi i}{2L}(2m+1)) \;\; i = 0, 1, \ldots, C-1 \tag{3.13}$$

En los experimentos se obtuvieron C =13 coeficientes sin incluir el coeficiente $c[0]$ que representa aproximadamente el logaritmo de la energía de la señal enventanada.

La aplicación de la DCT, en vez de la IFFT (transformada inversa de fourier) como se hace con el cepstrum, reduce el número de coeficientes necesarios para aproximar una señal de valores reales, debido a una propiedad de la misma conocida como compactación de la energía que explota la redundancia de dicha señal con variable en el dominio de los números reales (como ocurre con los coeficientes de energía). Además, desde el punto de vista computacional la DCT es más eficiente que la IFFT.

En la figura de a continuación, se resumen todos los pasos seguidos para la obtención de los parámetros MFCC:

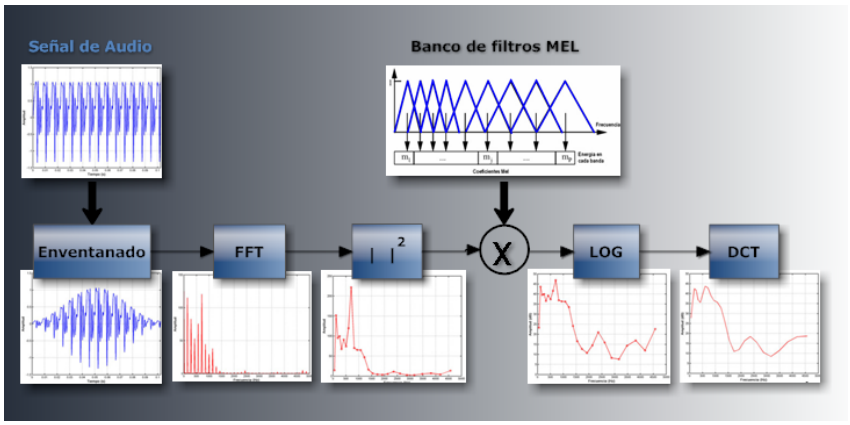

Figura 3.7: Pasos seguidos para la obtención de los parámetros MFCC.

3.2. SDC (Shifted Delta Cepstral) y SDC-R (SDC Regresivos)

Los coeficientes SDC aparecieron por primera vez en un a articulo de 1994 de B.Bielefeld [18]. Sin embargo, no sería hasta casi después de una década, en el año 2002, cuando un trabajo de T. Carrasquillo et Al [17] mostrará las bondades de los mismos en los sistemas de identificación de idioma. Desde entonces, los parámetros SDC se han venido usando con gran profusión en casi todos los sistemas acústicos que conforman el estado del arte en reconocimiento idiomático. De hecho, en la evaluación de idioma de NIST 2003 se constató que el empleo de coeficientes SDC puso a los sistemas acústico al mismo nivel que los PRLM.

Desde el punto vista formal los parámetros SDC son un tipo particular de coeficientes de velocidad (también llamados delta). El término velocidad se refiere a la primera derivada de los parámetros MFCC (o cualquier otra parametrización análoga) respecto del tiempo.

Se especifican por medio de 4 parámetros: $N - d - P - k$, cuyo significado es el siguiente:

1. N : número de coeficientes que componen el vector MFCC en cada ventana de tiempo (frame).
2. d : distancia superior e inferior respecto al frame central sobre el que se calcula la diferencia de los vectores MFCC.
3. P : desplazamiento entre bloques delta consecutivos.
4. k : número de bloques delta que se concatenan para formar el vector SDC final.

Elegidos estos parámetros los SDC se calculan simplemente aplicando la siguiente expresión:

$$\Delta C_n(t,i) = C_n(t+iP+d) - C_n(t+iP-d) \tag{3.14}$$
$$0 \leq n \leq N-1 0 \leq i \leq K-1,$$

donde $C_n(t)$es la n-ésima componente del vector MFCC calculado en el instante de tiempo (frame) t .

La parametrización empleada a lo largo de todos los experimentos fue la 7-1-3-7. Por consiguiente, el vector SDC tendrá 49 componentes.

Agregando los coeficientes SDC a los coeficientes MFCC estáticos, estamos incorporando a nuestro clasificador información temporal útil de la señal de voz como la de coarticulación de fonemas característica de cada idioma, mejorando así el rendimiento final del sistema.

En la siguiente figura se ilustra el procedimiento seguido para obtener los parámetros SDC 7-1-3-7.

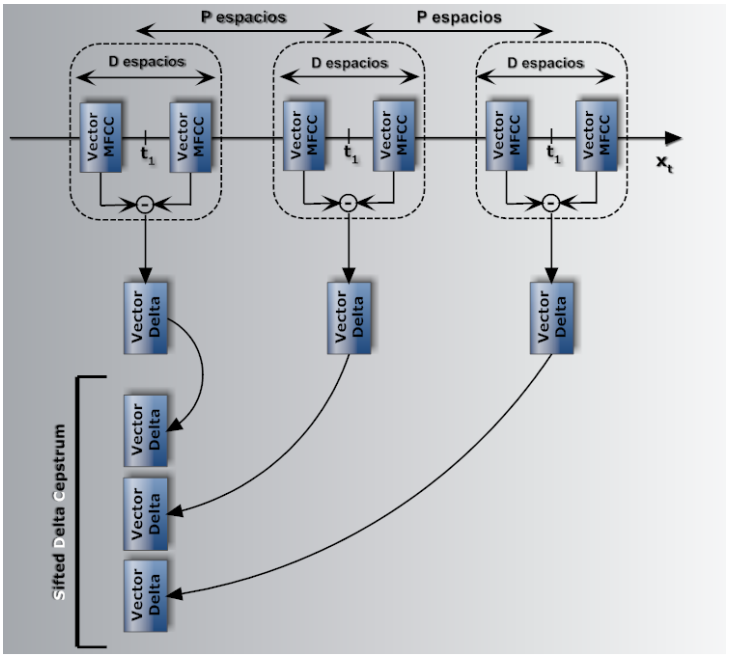

Figura 3.8: Diagrama de obtención de los parámetros SDC.

Motivado por los buenos resultados obtenidos en [11], los parámetros SDC's utilizados en los primeros experimentos del proyecto se calcularon siguiendo la siguiente formula de regresión:

$$\Delta c(t+iP) = \frac{\sum_{d=-D}^{D} dc(t+iP+d)}{\sum_{d=-D}^{D} d^2} \tag{3.15}$$

La parametrización empleada fue la misma que en [11], esto es, la 8-3-3-7.

Aunque los resultados en un conjunto reducido de audio obtenidos con esta parametrización fueron mejores que los SDC's clásicos 7-1-3-7, no se generalizo en posteriores experimentos empleando mucho mas audio y un corpus ligeramente distinto. Sin embargo, hay que ser conscientes que solo un estudio mas exhaustivo debería evidenciar cual parametrización es superior bajo determinadas condiciones (por ejemplo con mas o menos disponibilidad de audio para entrenar los modelos) y esto escapa a los objetivos de este proyecto.

3.3. Técnicas de compensación de canal

Uno de los mayores retos a los que se enfrenta el reconocimiento de idioma, es la variación que existe entre el audio utilizado para entrenar los modelos y las condiciones reales del audio de evaluación (variabilidad inter-sesión, donde una sesión es una grabación de audio en particular). En concreto, estas variaciones se deben a que se usan distintos locutores (variabilidad inter-locutor), distintos tipos de transductores, distintos canales de transmisión (líneas terrestres, celulares, VoIP), ruido ambiente, condiciones ambientales diferentes de grabación (donde puede haber mas o menos eco, por ejemplo) etc. Todos estos factores contribuyen a reducir el rendimiento de los sistemas y deben ser mitigados en lo posible por las técnicas que comentamos a continuación.

Por último, decir que todas las técnicas de compensación descritas a continuación han sido implementadas en código fuente (generalmente usando Matlab, por su sencillez de programación y ser multiplataforma).

3.3.1. Niveles de aplicación

Los efectos del canal pueden ser compensados en tres niveles, a saber:

- A nivel de parámetros: se modifican los vectores de características antes realizar el entrenamiento y evaluación. Las técnicas RASTA, CMS, CMVN, feature warping trabajan a este nivel eliminando los efectos lineales convolucionales del canal.
- A nivel de modelos: durante el proceso de verificación, los parámetros del modelo (típicamente un GMM) se modifican para adaptarse a las condiciones del canal del fichero de evaluación. Eigenchannel compensation (factor analisys) trabaja a este nivel.
- A nivel de puntuaciones (en la evaluación): los scores son modificados para eliminar efectos del canal. Las técnicas de normalización H-NORM, Z-NORM, T-NORM, etc., pertenecen a este nivel.

3.3.2. RASTA

El objetivo de RASTA (**R**el**A**tive **S**pec**T**r**A**l) es eliminar los efectos lineales de variación lenta o de baja frecuencia, que el canal de transmisión introduce en el audio utilizado para entrenar y evaluar los sistemas. Para aplicar RASTA, se supondrá que se disponen de de los primeros N coeficientes cepstrales (en los experimentos se utilizo N=12 coeficientes MFCC's, $c_1(t), c_2(t), \cdots, c_i(t), \cdots, c_{12}(t)$, omitiendo el coeficiente de energía c_0) .Consideremos cada coeficiente i-ésimo $c_i(t)$, como un flujo de valores (uno por cada frame) en la variable t (número de frame) . Pues bien, el objetivo de RASTA es pasar dicho stream de valores cepstrales a través

de un filtro $h(t)$ que consiga eliminar las componentes de baja frecuencia y algunas de las de alta frecuencia (filtro paso banda) que afectan negativamente al rendimiento del sistema.

De este modo los coeficientes cepstrales iniciales estarán relacionados con los coeficientes cepstrales procesados con RASTA por la ecuación:

$$c_i^{RASTA}(t) = h(t) * c_i(t)^1 \tag{3.16}$$

Se ha empleado, como suele ser usual en RASTA, un filtro de cuarto orden IIR con función de transferencia:

$$H(z) = 0{,}1\frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{z^{-4}(1 - 0{,}98z^{-1})} \tag{3.17}$$

Este filtro se ha escrito en Matlab empleando la función build-in filter.

3.3.3. CMS

El objetivo de **C**cepstral **M**ean **S**subtraction es parecido al de RASTA (eliminar las componentes de baja frecuencia que el canal introduce en el audio) y de hecho puede considerarse un caso particular del mismo. Para implementarlo simplemente se calcula el valor medio de c(n) (donde n es el número de coeficiente cepstral) de todos los frames del fichero (supondremos que tiene N frames) y se resta al coeficiente cepstral (en nuestro caso MFCC) correspondiente:

$$c_i^{CMS}[t] = c_i[t] - \frac{1}{T}\sum_{t=1}^{T} c_i[t] \quad i = 1, \cdots, D, \tag{3.18}$$

donde t representa el número de frame de un total de T, y el índice i la componente i-ésima del vector de características MFCC de dimensión D.

3.3.4. CMVN

CMVN (**C**epstral **M**ean **V**ariance **N**ormalization) es igual que CMS salvo que a parte de restarle la media se divide por la varianza. Esto es:

$$c_i^{CMVN}[t] = \frac{c_i[t] - \mu_i}{\sigma_i^2} \tag{3.19}$$

$$\mu_i = \frac{1}{T}\sum_{t=1}^{T} c_i[t] \tag{3.20}$$

$$\sigma_i^2 = \frac{1}{T-1}\sum_{t=1}^{T}(c_i[t] - \mu_i)^2 \tag{3.21}$$

$$i = 1, \cdots, D \tag{3.22}$$

[1] El simbolo * denota la operación de convolución

3.3.5. Feature Warping

Feature warping [11][13] es una técnica muy similar a la técnica conocida en tratamiento de imágenes como ecualización de histogramas, que se realiza con la luminancia (intensidad) de los píxeles de una imagen. Consideremos un fichero de audio compuesto por T vectores (uno por cada frame), cada uno a su vez compuesto por sendos coeficientes cepstrales $c(n)$ (en nuestro caso 12 coeficientes MFCC $c(n)$, $n = 1, 2, \ldots, 12$) y, consideremos también cada coeficiente n-ésimo como un flujo de valores independiente, esto es, $c(n)^i, i = 1, 2, ..., T$. Ahora calculemos el histograma de $c(n)^i$. Dicho histograma será una aproximación discreta a la función de densidad de probabilidad de dicho flujo de valores, que denotaremos por $f(y)$. El objetivo de Feature Warping es mapear dicha densidad de probabilidad a una función de densidad objetivo, que denotaremos por $h(z)$, de modo que se eliminen los efectos que el canal y el ruido introdujeron en la función de distribución original (antes de añadir los efectos del canal y el ruido).

El mapeado del valor cepstral $c(n) = q$ al valor mapeado m (warped), se hace de forma que se conserve la probabilidad acumulada, esto es:

$$\int_{y=-\infty}^{q} f(y)dy = \int_{z=-\infty}^{m} h(z)dz \tag{3.23}$$

Normalmente la función de distribución objetivo es un función gaussiana normal (media cero y varianza unidad).

En la siguiente figura se detalla el proceso de mapeado o warping que describiremos a continuación:

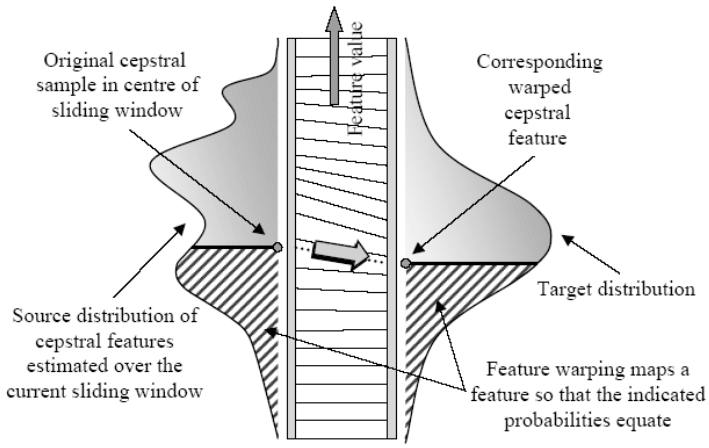

Figura 3.9: Proceso de mapeado (warping) de los coeficientes cepstrales.Figura extraida de [12].

El proceso es el siguiente. Se comienza obteniendo los valores de una ventana de tamaño N (en nuestro caso $N = 300$, lo que equivale a 3 segundos de audio con un frame rate de 10 ms) y se ordenan en orden descendente. Para determinar a que valor se mapea el coeficiente cepstral n-ésimo que se encuentra en el centro de la ventana se calcula el ranking R que ocupa dicho coeficiente dentro de la lista de valores ordenados de la ventana (al valor mas positivo de la ventana le corresponde un ranking de 1 y al mas negativo un ranking de N). Este ranking R se usa como índice de una tabla de lookup que contiene el valor mapeado (warped).

La tabla de lookup para realizar el mapeo se calcula antes de realizar el proceso de parametrización usando la siguiente ecuación:

$$\frac{N + \frac{1}{2} - R}{N} = \int_{z=-\infty}^{m} h(z)dz,$$

donde **h(z)** es la función de densidad de probabilidad objetivo, que en nuestro caso es un función de densidad normal (gaussiana) ,y **m** el valor mapeado (warped) que corresponde al índice R de la tabla de lookup, esto es**,**

$$m = \text{norminv}(\frac{N + \frac{1}{2} - R}{N}) \tag{3.24}$$

La función $h(z)$ tiene la forma:

$$h(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \tag{3.25}$$

En realidad la ecuación anterior es una aproximación a la forma continua:

$$\int_{y=-\infty}^{q} f(y)dy = \int_{z=-\infty}^{m} h(z)dz, \tag{3.26}$$

donde $f(y)$ es la función de densidad original de los datos.

En las figuras de a continuación, se puede ver el histograma del primer coeficiente cepstral c_1 antes y después de aplicar feature warping.

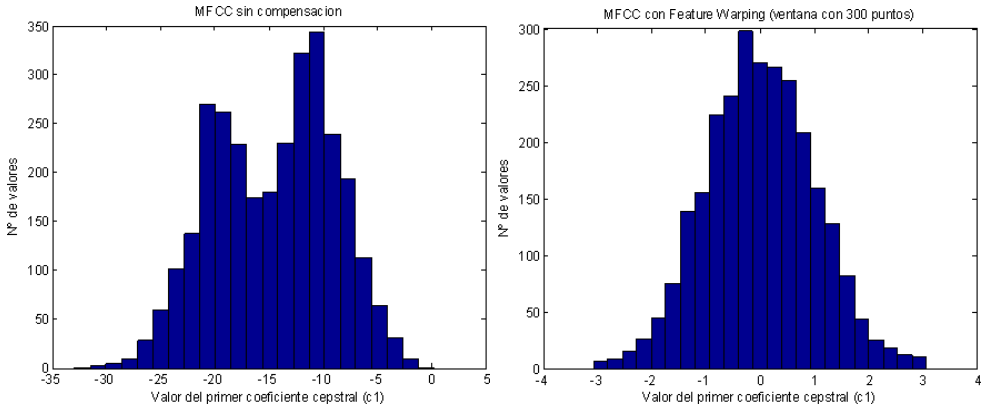

Figura 3.10: Histograma del coeficiente cepstral MFCC c_1 antes (izquierda) y después de aplicar feature warping.

3.3.6. T-Norm y Z-Norm

Las técnicas descritas en secciones anteriores intentaban compensar las variaciones del canal normalizando los vectores de características, en tanto que T-Norm y Z-Norm lo hacen escalando las distribuciones de las puntuaciones o scores. Ambas técnicas tienen su origen en el campo del reconocimiento de locutor, en donde han demostrado un buen desempeño. Es de esperar, que por la similitud con el reconocimiento de idioma también este se vea beneficiado de su aplicación.

Para aplicar T-Norm (Test Normalization) al score obtenido de enfrentar el fichero de test O al idioma objetivo representado por el modelo λ , se debe realizar la siguiente transformación:

$$S(O|\lambda)_{Z-Norm} = \frac{S(O|\lambda) - \mu_O}{\sigma_O} \tag{3.27}$$

$$\mu_o = \frac{1}{N-1} \sum_{\lambda' \neq \lambda} S(O|\lambda') \tag{3.28}$$

$$\sigma_o = \sqrt{\frac{1}{N-2} \sum_{\lambda' \neq \lambda} (S(O|\lambda') - \mu_o)^2}, \tag{3.29}$$

siendo N el número de idiomas totales. Al conjunto representado por los modelos de impostores $\lambda' \neq \lambda$ se le denomina cohorte.

Z-Norm se aplica de forma análoga, mas concretamente dado un idioma objetivo representado por el modelo λ, el score Z-normalizado $S(O|\lambda)_{Z-Norm}$se obtendrá de la siguiente forma:

$$S(O|\lambda)_{Z-Norm} = \frac{S(O|\lambda) - \mu_\lambda}{\sigma_\lambda} \tag{3.30}$$

$$\mu_\lambda = \frac{1}{M-1} \sum_{O' \notin \lambda} S(O'|\lambda) \tag{3.31}$$

$$\sigma_\lambda = \sqrt{\frac{1}{M-2} \sum_{O' \notin \lambda} (S(O'|\lambda) - \mu_\lambda)^2}, \tag{3.32}$$

siendo M el número de ficheros escogidos del conjunto de desarrollo que no contienen el idioma objetivo. Esto es, el score $S(O|\lambda)$ es normalizado por la media y desviación típica de la distribución de puntuación de ficheros de impostores del idioma objetivo λ.

Comparando la expresión de $S(O|\lambda)_{T-Norm}$ con la de $S(O|\lambda)_{Z-Norm}$, vemos que mientras que con T-Norm es el fichero de test el que se enfrenta a un conjunto de modelos de impostores para realizar la normalización, en Z-Norm es el modelo el que se enfrenta a un conjunto de ficheros de test de impostores. Idealmente, después de aplicar Z-Norm la distribuciones de scores de impostores de todos los idiomas deberían tener media cero y varianza unidad.

Un problema de Z-Norm, es que puede degradarse considerablemente, al punto de ser innecesario, si el audio de entrenamiento usado para estimar las medias y desviaciones típicas de los scores de ficheros de impostores, difiere en gran medida del audio de evaluación. Por esta razón durante el proyecto, salvo en un experimento, se ha empleado T-Norm. Si por el contrario el audio de entrenamiento y evaluación son parecidos entonces Z-Norm debería incrementar notablemente el rendimiento.

El uso de T-Norm no se limita solamente a compensar posibles efectos del canal, si no que también permite homogeneizar las puntuaciones de sistemas de reconocimiento diferentes, de forma que sea posible fusionarlos para obtener sistemas mejores que cualquiera de los componentes. Igualmente, Z-Norm permite elegir un umbral común para todos los idiomas involucrados en la tarea de verificación.

3.3.7. Eigenchannel Compensation

Eigenchannel compesation (versión simplificada de Factor Analysis) es la técnica mas compleja que se ha implementado y al mismo tiempo la de mejor rendimiento. Puede ser aplicada en el dominio de los parámetros (forma implementada) o en el dominio de los modelos. Esta técnica ha sido la novedad en la última convocatoria de NIST LRE 2009.

Para aplicar esta técnica se deben de obtener los supervector de todos los ficheros de entrenamiento, siguiendo el proceso descrito en la sección anterior "GMM-SVM". Tal supervector será un vector de dimensión MD, siendo M el número de mezclas y D la dimensión del vector de características (en la implementación $D = 56$ componentes). A diferencia del sistema GMM-SVM, el supervector se formará concatenando únicamente los vectores de medias (no se normalizará por las varianzas).

Para cada supervector se trata de identificar las direcciones en las que este es afectado por las variaciones del canal. Estas direcciones, conocidas como eigenchannels, se definen por las columnas de una matriz V de dimensiones $MD \times R$, siendo R el número de autocanales (en los experimentos $R = 50$). La matriz V esta formada por R autovectores de medias dentro de la matriz de covarianzas, y se obtiene mediante análisis PCA (Principal Component Analisys) de los supervector de los ficheros de entrenamiento.

Una vez que los autocanales son identificados, el supervector que representa el GMM de un modelo de idioma obtenido por MAP del UBM, es adaptado a un fichero de test en particular, desplazándolo en la dirección determinada por los autovectores, ajustándose de esta forma a las condiciones del canal presentes en el susodicho fichero de test. Matemáticamente, esto equivale a encontrar los denominados channel factors $\vec{x}$, que maximicen el siguiente criterio ML:

$$p(O|\vec{s} + V\vec{x}), \tag{3.33}$$

donde $\vec{s}$ es el supervector que representa el modelo a ser adaptado, y $p(O|\vec{s} + V\vec{x})$ es la probabilidad de que el fichero de test $o(t), t = 1, \cdots, T$ condicionada al supervector del modelo adaptado (compensado). Se puede demostrar, que una solución que maximiza el criterio anterior esta dada por:

$$x = A^{-1} \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_m(t) V_m^T \Sigma_m^{-1} \left(o(t) - \mu_m\right) \tag{3.34}$$

$$A = \sum_{m=1}^{M} \left(\sum_{t=1}^{T} \gamma_m(t) \right) V_m^T \tag{3.35}$$

$$\Sigma_m^{-1} V_m \gamma_m(t) = \frac{c_m N(o_t(t); \mu_m, \sigma_m^2)}{\sum_{j=1}^{M} c_j N(o_t(t); \mu_j, \sigma_j^2)}, \tag{3.36}$$

donde V_m se corresponde con la submatriz $D \times R$ de V, correspondiente la m-éisma mezcla componente, $\gamma_m(t)$ es la probabilidad de ocupación de la mezcla m-ésima en el instante t, y, finalmente, Σ_m y $\mu_m$la matriz diagonal de varianzas y el vector de medias, respectivamente, de la mezcla m-ésima. Las probabilidades de ocupación, $\gamma_m(t)$, se calculan usando el modelo UBM.

Para obtener la matriz eigenchannel V, se siguieron los siguientes pasos:

1. Se entrenó un UBM usando el audio de todos los idiomas.

2. Para cada fichero de vectores de características, se obtiene un GMM por adaptación MAP del UBM.

3. Se forma un supervector concatenando la medias de las mezclas que componen el GMM anterior (no se normalizan por las varianzas).

4. Se seleccionaron 80 ficheros (de duración 30 minutos) por idioma de la base de datos Callfriend. A partir de aquí es donde se empieza a aplicar la técnica PCA (Principal Component Analisys).

5. A los 80 supervector de cada idioma se les calcula la media. El supervector de medias de cada idioma se les resta a los supervectors del idioma correspondiente. Los supervector resultantes se disponen en columnas para formar la matriz S de dimensión $MD \times 80$.

6. Finalmente, la matriz V se forma seleccionando los R (número de canales) autovectores de la matriz de covarianza $(1/80)SS^T$ con los mayores autovalores. Debido a la alta dimensionalidad de la matriz $(1/80)SS^T$, en vez de calcular los autovectores de la misma, se calculan los autovectores V' de la matriz $(1/80)S^TS$ (de dimensión 80×80 en vez de $MD \times MD$). De esta forma, $V = SV'$.

La forma implementada de eigenchannel, no se ha realizado a nivel de modelos,si no modificando los vectores de características. Una de las ventajas de esta última aproximación es que nos permite aprovecharnos de la ventajas de la compensación con cualquier tipo de clasificador (GMM-MMI, SVM,etc).

Para cada fichero de parámetros $o(t)$ con $\mathrm{t} = 1, \cdots \mathrm{T}$, se compensa del siguiente modo:

$$\hat{o}(t) = o(t) - \sum_m \gamma_m(t) V_m x^{(i)} \quad t = 1, \cdots \mathrm{T}, \tag{3.37}$$

donde $\hat{o}(t)$ es el fichero compensado, $\gamma_m(t)$, la probabilidad de ocupación de la mezcla m-ésima calculado usando el UBM, y $\vec{x}$ los channel factors del fichero. La sumatoria anterior solo se realizo con las 5 mezclas mas pesadas del UBM.

El proceso anterior, se hace tanto para los ficheros de evaluación como los de entrenamiento. A continuación se utiliza cualquier clasificador (GMM-MMI, SVM, etc) utilizando estos parámetros. En la parte de experimentos veremos los increíbles resultados que proporciona esta técnica.

3.3.8. Detector de silencios

Se ha empleado un detector de silencios basado en la potencia media espectral de la señal de audio. Para una ventana de datos x, la potencia media espectral se calcula como:

$$P_{avg}(x) = \frac{1}{N}\left[\sum_{n=1}^{N} |F_n(x)|^2\right], \tag{3.38}$$

donde $F_n(x)$ es el n-ésimo termino de la transformada de Fourier de x, y N el tamaño de dicha transformada.

Se emplea un umbral T para determinar si el frame se considera un silencio. Si $P_{avg}(x) < T$ entonces se toma como silencio. El umbral empleado ha sido de $T = 0{,}00056234$, que equivale a *-65 dB*.

En las figuras de a continuación, se muestran el espectrograma (con las regiones consideradas como audio sin silencios) y la potencia de un fichero de audio típico de 30 segundos (obtenido de los ficheros de evalset de la evaluación de idioma de NIST 2003).

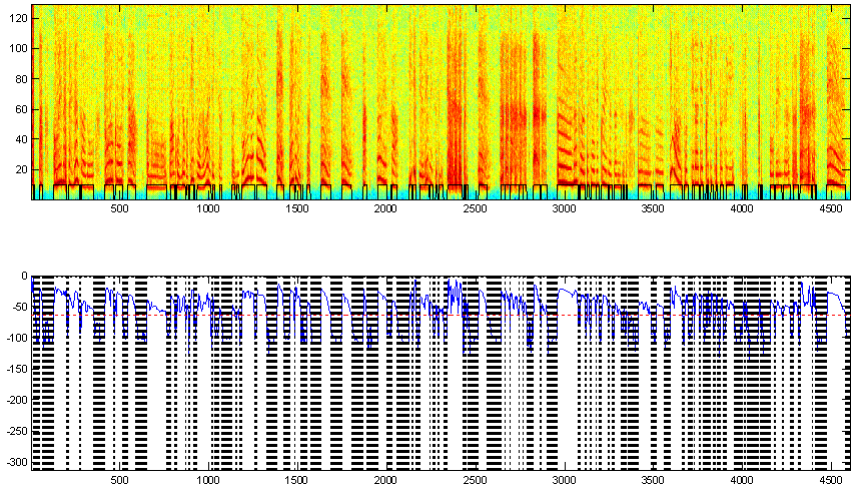

Figura 3.11: Espectrograma (arriba) y potencia media espectral de un fichero de audio extraído de la evaluación de idioma de NIST 2003.

4

El paradigma de entrenamiento MMI

Este capitulo esta dedicado exclusivamente al estudio del paradigma de entrenamiento discriminativo MMI. En el se verá por qué el entrenamiento discriminativo es más adecuado que el clásico entrenamiento generativo cuando se aplica al área de reconocimiento de idioma, junto con los algoritmos computacionales que existen para su implementación. Comenzaremos definiendo el concepto de máxima información mutua.

4.1. El criterio de máxima información mutua

El uso del criterio MMI [24] fue aplicado a problemas de reconocimiento de patrones, en especial para reconocimiento de voz, originariamente por Bahl et al, como una alternativa al criterio ML. El objetivo del criterio MMI aplicado a reconocimiento de idioma, es encontrar un conjunto de parámetros que maximicen la cantidad de información mutua entre los segmentos de audio y el idioma al que pertenecen. El concepto de MMI deriva directamente de la teoría de la información de Shanon, que pasamos a describir a continuación.

En términos generales, se define la información mutua entre dos variables aleatorias, que para simplificar supondremos discretas, como la cantidad de información que una variable posee sobre la otra. Sean $\boldsymbol{X}$ e $\boldsymbol{Y}$ dichas variables aleatorias y, $\boldsymbol{x}$ e $\boldsymbol{y}$, sus respectivas salidas (realizaciones), la información mutua entre $\boldsymbol{X}$ e $\boldsymbol{Y}$, denotada por *I(X;Y)*, es :

$$I(X;Y) = H(X) - H(X|Y), \tag{4.1}$$

donde *H(X)* es la entropía de $\boldsymbol{X}$ y $H(X|Y)$ la entropía condicional de $\boldsymbol{X}$ dada $\boldsymbol{Y}$, definidas como:

$$H(X) = -\sum_{x \in X} P(x) \log P(x) \tag{4.2}$$

$$H(X|Y) = -\sum_{x \in X} \sum_{y \in Y} P(x,y) \log P(x|y) \tag{4.3}$$

1. $H(X)$ es la cantidad de incertidumbre (o de información) de la variable aleatoria $\boldsymbol{X}$. Si log esta en base 2, entonces *H(X)* es el número mínimo de bits necesarios para codificar todas las posibles realizaciones (salidas) de la variable aleatoria $\boldsymbol{X}$ supuesta esta discreta.
2. $H(X|Y)$ es la entropía condicional, o la cantidad de incertidumbre (información) de $\boldsymbol{X}$ después de conocer $\boldsymbol{Y}$.

De lo anterior se deduce que *I(X; Y)* mide la reducción en la cantidad de incertidumbre de $\boldsymbol{X}$ después de conocer $\boldsymbol{Y}$. De las ecuaciones de arriba, *I(X;Y)* puede ser expresada como:

$$\begin{aligned} I(X;Y) &= -\sum_{x\in X} P(x)\log P(x) - \left(-\sum_{x\in X}\sum_{y\in Y} P(x,y)\log P(x|y)\right) = \\ &= -\sum_{x\in X}\sum_{y\in Y} P(x,y)\log P(x) + \sum_{x\in X}\sum_{y\in Y} P(x,y)\log P(x|y) = \\ &= \sum_{x\in X}\sum_{y\in Y} P(x,y)\log\frac{P(x|y)}{P(x)} \end{aligned} \tag{4.4}$$

Expresada de esta forma, podemos ver que *I(X;Y)* es la media, tomada sobre el espacio muestral de X e Y, de la variable aleatoria:

$$I(x;y) = \log\frac{p(x|y)}{p(x)} \tag{4.5}$$

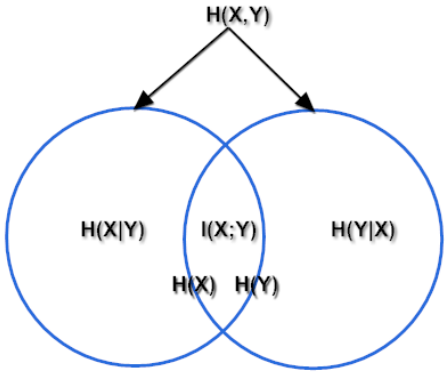

Figura 4.1: Diagrama de Información (o de Venn) mostrando las variables involucradas. El círculo de la izquierda representa las realizaciones de la variable.

El criterio MMI se deriva directamente de la ecuación de arriba. Mas concretamente, dado un conjunto de R segmentos de audio disponibles para el entrenamiento, $O = \{O_1, ..., O_r, ..., O_R\}$y sus correspondientes categorías (en nuestro caso, serán los idiomas) $C = \{C^1, ..., C^r, ..., C^R\}$ donde $C^r \in \{C_1, ..., C_L\}$. O_r es el r-ésimo segmento de entrenamiento y C^rla correspondiente categoría (idioma) de O_r, la información mutua $I(O_r; C^r)$ entre el segmento de entrenamiento O_r y su correspondiente categoría C^r, será (empleando la ecuación anterior):

$$I(O_r;C^r) = \log\frac{P(O_r;C^r)}{P(O_r)} = \log P(O_r;C^r) - \log P(O_r) = \log P(O_r|C^r) - \log\sum_{i=1}^{L} P(O_r|C_i)P(O_i)$$

Supondremos que la función de probabilidad condicional P(O|C) puede ser modelada por una mezcla de Gaussianas multidimensionales (GMM) cuyos parámetros (pesos de las mezclas, medias y varianzas) denotaremos por λ. La idea del criterio MMI es estimar un conjunto de parámetros λ que maximicen la información mutua entre los segmentos de entrenamiento O y sus categorías (idioma) correctas. Esto es, lo que queremos es maximizar la función objetivo MMI, f_{MMI}, de la información mutua media para todos los segmentos de entrenamiento definida como:

$$f_{MMI}(\lambda) = \frac{1}{R}\sum_{r=1}^{R}\log\frac{P(O_r|C^r)}{\sum_{i=1}^{L}P(O_r|C_i)P(C_i)} = \frac{1}{R}\sum_{r=1}^{R}(\log P_\lambda(O_r|C^r) - \log\sum_{i=1}^{L}P_\lambda(O_r|C_i)P(C_i))$$

Si suponemos que la probabilidades a priori $P(C_i)$ son la mismas para todos los idiomas el conjunto de parámetros de los modelos Φ que maximiza la función anterior coincide con el de la función :

$$f_{MMI}(\lambda) = \sum_{r=1}^{R}\log\frac{P(O_r|C^r)}{\sum_{i=1}^{L}P(O_r|C_i)} = \sum_{r=1}^{R}(\log P_\lambda(O_r|C^r) - \log\sum_{i=1}^{L}P_\lambda(O_r|C_i)),$$

que queda en la forma final que se utilizó para optimizar.

Con la maximización de f_{MMI}, incrementamos la diferencia entre $P_\lambda(O_r|C^r)$ y la densidad de probabilidad $P_\lambda(O_r) = \sum_{i=1}^{M}P_\lambda(O_r|C_i)P(C_i)$, a diferencia del criterio de máxima verosimilitud (ML) donde solo intenta maximizar la probabilidad condicional $P_\lambda(O_r|C^r)$.

En otras palabras,en el entrenamiento MMI no solo intenta maximizar la probabilidad condicional de los segmentos de entrenamiento dados sus idiomas correspondientes (categorías), sino también minimizar al mismo tiempo la probabilidad condicional de todos los idiomas posibles que compiten con el correcto.

El conjunto de parámetros λ,se elige de forma que mejore la discriminación entre los segmentos de audio utilizados para el entrenamiento. Por esta razón el criterio de maximización MMI se considera como un método de entrenamiento discriminativo. Con el entrenamiento discriminativo nuestro interés se concentra en encontrar las fronteras que separan a las distintas categorías (idiomas, clases) en vez de aumentar solamente la probabilidad a posteriori de una categoría independientemente de las demás.

De esta última consecuencia se desprende que con el criterio MMI, a diferencia del método usual ML, donde el objetivo es modelar la densidad de probabilidad no se requiere que la forma funcional de la distribución de probabilidad de los modelos sea correctamente especificada. Se ha demostrado matemáticamente que , bajo ciertas circunstancias, el uso del método de entrenamiento MMI con una forma incorrecta de función de probabilidad converge a un resultado óptimo si existen suficientes datos de entrenamiento, mientras que no ocurre lo mismo con el entrenamiento ML.

El principal inconveniente del método MMI, es que la maximización de la función objetivo MMI requiere muchos mas cálculos que maximizar por ML. Para cada segmento de entrenamiento, necesitamos calcular la probabilidad condicional respecto al idioma al que pertenece y para todos lo restantes. Otro de los problemas importantes a los que se enfrenta el criterio MMI, es que los algoritmos para optimizar la función objetivo MMI no son tan eficientes como el conocido Baum-Welch para el criterio ML, por lo que se requiere un mayor número de iteraciones para disponer de una buena convergencia al valor real de la función objetivo.

4.2. Definición del problema de optimización MMI

Cuando las funciones de distribución de probabilidad de los idiomas,$P_\lambda(O|C)$, se modelan como una mezcla de funciones de probabilidad multidimensionales gaussianas (GMM) y se emplea el criterio MMI para entrenar sus parámetros. Al GMM resultante lo denominamos discriminativo (DGMM).

La forma funcional de una de las mezclas del GMM es:

$$p(\vec{x}|\lambda) = \frac{w}{\sqrt{(2\pi)^D\,|\Sigma|}} e^{\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma^{-1}(\vec{x}-\vec{\mu})\right]}, \tag{4.6}$$

siendo $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}$, el vector de características (con la parametrización correspondiente, MFCC, SDC, LPC, etc); $\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_1^2 & 0 & 0 \\ 0 & 0 & \sigma_1^2 & 0 \\ 0 & 0 & 0 & \sigma_1^2 \end{bmatrix}$, la matriz de covarianza supuesta diagonal; $\vec{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_D \end{bmatrix}$, el vector de medias y w el peso de la mezcla.

Por λ denotamos los parámetros de la mezcla, esto es, $\lambda = \{w, \mu, \Sigma\}$.

Debido a que la matriz de covarianza es diagonal la fórmula de la función de distribución puede simplificarse después de algunos cálculos. A saber, teniendo en cuenta que el determinante de una matriz diagonal es el producto de los elementos de la diagonal principal, esto es:

$$|\Sigma| = \prod_{i=1}^{D} \sigma_i^2 \tag{4.7}$$

y que la inversa de una matriz diagonal es igual a sus elementos diagonales invertidos:

$$\Sigma^{-1} = \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & 0 & 0 \\ 0 & \frac{1}{\sigma_2^2} & 0 & 0 \\ 0 & 0 & \frac{1}{\sigma_3^2} & 0 \\ 0 & 0 & 0 & \frac{1}{\sigma_4^2} \end{bmatrix}, \tag{4.8}$$

la fórmula de la función de densidad de probabilidad, queda finalmente de la siguiente forma:

$$p(\vec{x}|\ \lambda) = \frac{1}{\sqrt{(2\pi)^D \prod_{i=1}^{D} \sigma_i^2}} e^{\sum_{i=1}^{D} -\frac{(x_i-\mu_i)^2}{2\sigma_i^2}} \tag{4.9}$$

Ahora se definirá el problema de optimización que se deberá resolver para poder implementar el criterio MMI en reconocimiento de idioma.

4.3. Definición del problema de optimización

Pasamos a describir los elementos que integran nuestro problema:

- En total se disponen de R segmentos de audio $O = \left\{O_1^{T_1}, ..., O_r^{T_r}, ..., O_R^{T_R}\right\}$ para el entrenamiento. A su vez cada segmento $O_r^{T_r}$ es una secuencia de T_r vectores de características, esto es, $O_r^{T_r} = \{o_{r,1}, ..., o_{r,t}, ..., o_{r,T_r}\}$. Su duración es de aproximadamente 30 segundos y se obtuvieron de las evaluaciones NIST de reconocimiento de idioma. Los vectores de características tienen 39 componentes como resultado de parametrizar el audio con 13 MFCC's, 13 ΔMFCC's y 13 ΔΔ MFCC's (coeficientes de velocidad y aceleración, respectivamente).

- De cada segmento se conoce la categoría (idioma) a la que pertenece, esto es, sabemos $C = \left\{C^1, ..., C^r, ..., C^R\right\}$. En total existen L categorías distintas, tantas como idiomas, esto es, $C^r \in \{C_1, ..., C_L\}$. Para la evaluación de Nist de 2003,por ejemplo, los idiomas fueron {spanish, english, arabic, farsi, french, german, japanese, korean, mandarin, russian, tamil, vietnamese }. Por tanto $L = 12$.

- Los idiomas son modelos con GMM's de M mezclas y parámetros $\lambda_i = \{w_{ij}, \mu_{ij}, \Sigma_{ij}\}, i = 1, ..., L; j = 1, ..., M$ .Las matrices de covarianza Σ son diagonales. La probabilidad de que un segmento $O_r^{T_r}$ pertenezca al modelo de idioma λ_i (esto es, a la categoría de idioma C_i) se calcula como (suponiendo que los vectores de características son independientes) :

$$p(O_r^{T_r}|\lambda_i) = \prod_{t=1}^{T_r} \sum_{j=1}^{M} \frac{w_{ij}}{\sqrt{2\pi \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk} - \mu_{ijk}\right)^2}{2\sigma_{ijk}^2}} \tag{4.10}$$

- Finalmente la función objetivo MMI que optimizaremos, y que ya hemos descrito en apartados anteriores, será, pues:

$$\begin{array}{l} f_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{P(O_r|C^r)}{\sum_{i=1}^{L} P(O_r|C_i)} = \sum_{r=1}^{R} \log \frac{P(O_r^{T_r}|\lambda^r)}{\sum_{i=1}^{L} P(O_r^{T_r}|\lambda_i)} = \\ = \sum_{r=1}^{R} (\log P(O_r^{T_r}|\lambda^r) - \log \sum_{i=1}^{L} P(O_r^{T_r}|\lambda_i)) \end{array} \tag{4.11}$$

- Si omitimos el denominador de la función f_{MMI} el criterio MMI se convierte en el conocido criterio ML (Maximum Likelihood). Esto es:

$$f_{ML}(\lambda) = \sum_{r=1}^{R} \log P(O_r^{T_r}|\lambda^r) \tag{4.12}$$

  Inicialmente los modelos son entrenados con este criterio antes de ser entrenados con el criterio MMI.

Entre los distintos métodos que existen para optimizar la función anterior, estudiaremos tres, dos basados en gradiente y un tercero derivado del utilizado para entrenar modelos ocultos de Markov (HMM) por ML, el algoritmo Baum-Welch pero modificado para ser aplicable a MMI. Es conocido por Baum-Welch Extendido. Pasamos a describir dichos algoritmos.

4.3.1. Algoritmos de optimización

4.3.1.1. Gradient-Descent (GD)

A diferencia del entrenamiento ML donde los parámetros de los modelos pueden ser entrenados eficientemente por el algoritmo Baum-Welch, este no puede ser aplicado para la optimización de la función objetivo MMI (después veremos una adaptación del Baum-Welch, el Extended-Baum-Welch, para MMI).Por esta razón, en una primera etapa, la investigaciones en entrenamiento MMI se concentraron en el uso de métodos basados en gradiente. Todos los métodos basados en gradiente intentan minimizar o maximizar la función objetivo iteratívamente, involucrando para ello el cálculo del gradiente que determinará la cantidad en que se verá modificado el parámetro correspondiente. Entre los métodos basados en gradiente, el método Gradient Descent (en adelante GD)[23] es el más simple (y por ello, también el de peor rendimiento) El termino *descent* suele usarse normalmente, incluso aunque en la práctica el algoritmo pueda ser aplicado a problemas de maximización donde el término *ascent* seria mas apropiado.

El procedimiento de optimización por GD es el siguiente. La optimización comienza con un modelo con parámetros iniciales λ. En cada iteración los parámetros son actualizados de acuerdo a la fórmula:

$$\tilde{\lambda} = \lambda + \varepsilon \nabla f_{MMI} \tag{4.13}$$

Los parámetros son modificados en la dirección especificada por el gradiente de la función objetivo, y la cantidad en que son modificados, es determinada por una constante ε, conocida como tasa de aprendizaje.

Aunque el algoritmo GD es en principio una técnica genérica que garantiza encontrar un máximo o mínimo local, su principal problema es su lenta tasa de convergencia, con la cual podría tomar hasta infinitas iteraciones para alcanzar el punto óptimo, y en especial en problemas donde el número de variables es muy elevado (como en un GMM con muchas mezclas y vectores de dimensiones elevadas).

Pasamos a describir ahora el calculo del gradiente de la función objetivo, que nos permita aplicar la ecuación $\tilde{\lambda} = \lambda + \varepsilon \nabla f_{MMI}$. Por ejemplo, para actualizar el valor del vector de medias de una de las mezclas de un modelo cualquiera, debemos realizar el siguiente calculo:

$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_D \end{bmatrix}^{t+1} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_D \end{bmatrix}^{t} + \varepsilon \begin{bmatrix} \frac{\partial F}{\partial \mu_1} \\ \vdots \\ \vdots \\ \frac{\partial F}{\partial \mu_D} \end{bmatrix}, \tag{4.14}$$

para lo cual deberemos saber calcular esas derivadas parciales de la función objetivo respecto del parámetro considerado.

Antes de nada empezaremos definiendo la función objetivo que se dedujo de apartados anteriores:

$$f_{MMI}(\lambda) = \sum_{r=1}^{R}(\log P(O_r^{T_r}|\lambda^r) - \log \sum_{i=1}^{L} P(O_r^{T_r}|\lambda_i)) \tag{4.15}$$

$$p(o_{rt}|\lambda_i) = \sum_{j=1}^{M} \frac{w_{ij}}{\sqrt{2\pi \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}} \tag{4.16}$$

Aunque podríamos intentar calcular el gradiente en la forma en que esta definida la función anterior, existe un problema de cara a su implementación. Debido a que $P(O_r^{T_r}|\lambda^r)$ puede llegar a ser extremadamente pequeño, incluso mas pequeño de lo que pueda representarse con un valor "double" en coma flotante, que equivale +1.7E-308. Por esta razón en vez de optimizar al nivel de segmento de audio, optimizaremos a nivel de frame. Esta claro que si optimizamos a nivel de frame también optimizaremos a nivel de fichero. Esto es, la función objetivo para optimizar por gradiente queda de la forma:

$$f_{MMI}(\lambda) = \sum_{r=1}^{R}\sum_{t=1}^{T_r}(\log P(o_{t,r}\lambda^r) - \log \sum_{i=1}^{L} P(o_{t,r}|\lambda_i))) \tag{4.17}$$

Una vez definida correctamente la función objetivo pasemos a calcular su gradiente respecto de los pesos, medias y varianzas de los modelos GMM's.

Comencemos con los pesos de las mezclas. El gradiente de $f_{MMI}(\lambda)$ respecto de w_{ij} (peso de la mezcla j-ésima del modelo i-ésimo), es:

$$\frac{\partial f_{MMI}(\lambda)}{\partial w_{ij}} = \sum_{r=1}^{R}\sum_{t=1}^{T_r}(\frac{1}{P(o_{t,r}\lambda^r)}\frac{\partial P(o_{t,r}\lambda^r)}{\partial w_{ij}} - \frac{1}{\sum_{i=1}^{L} P(o_{t,r}|\lambda_i)}\sum_{i'=1}^{L}\frac{\partial P(o_{t,r}|\lambda_{i'})}{\partial w_{ij}})) \tag{4.18}$$

Si $\lambda^r = \lambda_i$ la expresión anterior queda:

$$\begin{aligned}\frac{\partial f_{MMI}(\lambda)}{\partial w_{ij}} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}(\frac{1}{P(o_{t,r}\lambda^r)}\frac{1}{\sqrt{2\pi \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}} - \\ &- \frac{1}{\sum_{i'=1}^{L} P(o_{t,r}|\lambda_{i'})}\frac{1}{\sqrt{2\pi \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}))\end{aligned} \tag{4.19}$$

Si $\lambda^r \neq \lambda_i$ entonces:

$$\frac{\partial f_{MMI}(\lambda)}{\partial w_{ij}} = -\sum_{r=1}^{R}\sum_{t=1}^{T_r}\frac{1}{\sum_{i'=1}^{L} P(o_{t,r}|\lambda_{i'})}\frac{1}{\sqrt{2\pi \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}} \tag{4.20}$$

Podemos formular la ecuación anterior de forma mas concreta utilizando el símbolo de kronecker , a saber:

$\delta(r,i) = \begin{cases} 1, \lambda^r = \lambda_i \\ 0, \lambda^r \neq \lambda_i \end{cases}$, con lo que la derivada parcial queda:

$$\begin{aligned}
\frac{\partial f_{MMI}(\lambda)}{\partial w_{ij}} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{1}{P(o_{t,r}\lambda^r)}\frac{1}{\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\delta(r,i) - \\
&- \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\frac{1}{\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\Big) \qquad (4.21)\\
\frac{\partial f_{MMI}(\lambda)}{\partial \mu_{ijk}} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{1}{P(o_{t,r}\lambda^r)}\frac{\partial P(o_{t,r}\lambda^r)}{\partial\mu_{ijk}} - \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\sum_{i'=1}^{L}\frac{\partial P(o_{t,r}|\lambda_{i'})}{\partial\mu_{ijk}}\Big) = \\
&= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{1}{P(o_{t,r}\lambda^r)}\frac{w_{ij}\left(o_{rtk}-\mu_{ijk}\right)}{\sigma_{ijk}^2\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\delta(r,i) - \\
&- \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\frac{w_{ij}\left(o_{rtk}-\mu_{ijk}\right)}{\sigma_{ijk}^2\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\Big) \qquad (4.22)
\end{aligned}$$

Para las varianzas procedemos del mismo modo, a saber:

$$\begin{aligned}
&\tfrac{\partial f_{MMI}(\lambda)}{\partial\sigma_{ijk}^2} = \textstyle\sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{1}{P(o_{t,r}\lambda^r)}\frac{\partial P(o_{t,r}\lambda^r)}{\partial\sigma_{ijk}^2} - \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\sum_{i'=1}^{L}\frac{\partial P(o_{t,r}|\lambda_{i'})}{\partial\sigma_{ijk}^2}\Big) = \\
&= \textstyle\sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{1}{P(o_{t,r}\lambda^r)}\frac{w_{ij}}{\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\delta(r,i)\left(-\frac{1}{2\sigma_{ijk}^2}+\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2\sigma_{ijk}^2}\right) - \\
&- \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\frac{w_{ij}}{2\sigma_{ijk}^2\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\left(-\frac{1}{2\sigma_{ijk}^2}+\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2\sigma_{ijk}^2}\right)\Big)
\end{aligned} \qquad (4.23)$$

Resumiendo las fórmulas resultantes son:

$$\begin{aligned}
\frac{\partial f_{MMI}(\lambda)}{\partial w_{ij}} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{1}{\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\Big(\frac{1}{P(o_{t,r}\lambda^r)}\delta(r,i) - \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\Big)\Big)\\
\frac{\partial f_{MMI}(\lambda)}{\partial \mu_{ijk}} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{w_{ij}\left(o_{rtk}-\mu_{ijk}\right)}{\sigma_{ijk}^2\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\Big)\Big(\frac{1}{P(o_{t,r}\lambda^r)}\delta(r,i) - \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\Big)\\
\frac{\partial f_{MMI}(\lambda)}{\partial \sigma_{ijk}^2} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\Big(\frac{w_{ij}}{\sqrt{2\pi\prod_{k=1}^{D}\sigma_{ijk}^2}}e^{\sum_{k=1}^{D}-\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}\left(\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2\sigma_{ijk}^2}-\frac{1}{2\sigma_{ijk}^2}\right)\Big(\frac{1}{P(o_{t,r}\lambda^r)}\delta(r,i) - \\
&- \frac{1}{\sum_{i'=1}^{L}P(o_{t,r}|\lambda_{i'})}\Big)\Big)
\end{aligned}$$

Generalmente, el procedimiento GD es un método de minimización sin restricciones que necesita ser modificado para satisfacer las restricciones de la función de distribución del modelo

de idioma considerado. Estas restricciones son que la suma de los pesos de las mezclas del GMM deben sumar la unidad y las varianzas de las mismas deben ser números positivos.Para ello se realizara una transformación de los parámetros que implícitamente mantenga las restricciones durante el empleo del método GD. Los parámetros originales son actualizados a través de la transformación inversa del espacio de parámetros transformados al espacio de parámetros original. La transformación es realizada de modo que siempre se mantengan las restricciones de los parámetros originales. Algunas de estas transformaciones se enumeran a continuación:

1. Para valores de probabilidad que deben ser positivos y sumar la unidad, tales como los valores discretos de una función de probabilidad o los pesos de las mezclas de un modelo GMM, puede realizarse la siguiente transformación:

$$a_t = \frac{\exp(\tilde{a}_t)}{\sum_k \exp(\tilde{a}_k)} \tag{4.24}$$

2. Para la media μ y varianza (o las entradas de la diagonal de la matriz de covarianza) σ^2, se pueden emplear las siguientes transformaciones:

$$\mu = \tilde{\mu}\sigma \qquad \sigma^2 = e^{\tilde{\sigma}} \tag{4.25}$$

Después de las transformaciones, podemos calcular el gradiente con respecto a los parámetros transformados usando la regla de la cadena. Una vez que los parámetros transformados sean obtenidos por el método GD, podemos obtener los parámetros originales sin mas que sustituir en las fórmulas de transformación anteriores.

Debido a que solo actualizaremos las medias y varianzas, se empleara únicamente la segunda transformación. Por ejemplo para actualizar la componente *k-ésima* del vector de varianzas de la mezcla *j-ésima* del modelo de idioma *i-ésimo*, esto es, ${\sigma_{jk}^2}^i$ y asegurar que después de actualizarla con el método GD siga siendo positiva, en vez de realizar esto (lo cual no garantiza que el nuevo valor de la varianza siga siendo positivo) haríamos:

$$\tilde{\sigma_{jk}^2}^i = {\sigma_{jk}^2}^i - \varepsilon \frac{\partial J_{MMI}}{\partial {\sigma_{jk}^2}^i}, \tag{4.26}$$

utilizaremos una variable auxiliar h, tal que:

$$h_{jk}^i = \log {\sigma_{jk}^2}^i \tag{4.27}$$

y desarrollamos el método GD para la misma, de forma que:

$$\tilde{h}_{jk}^i = h_{jk}^i - \varepsilon \frac{\partial J_{MMI}}{\partial h_{jk}^i} \tag{4.28}$$

Aplicando la regla de la cadena, la ecuación anterior queda de la forma:

$$\tilde{h}_{jk}^i = h_{jk}^i - \varepsilon \frac{\partial J_{MMI}}{\partial {\sigma_{jk}^2}^i} \frac{{\sigma_{jk}^2}^i}{\partial h_{jk}^i} = h_{jk}^i - \varepsilon \frac{\partial J_{MMI}}{\partial {\sigma_{jk}^2}^i} e^{h_{jk}^i}. \tag{4.29}$$

Una vez obtenido el nuevo valor para h, recobramos al parámetro original, simplemente aplicando la transformación inversa:

$$\tilde{\sigma_{jk}^2}^{\,i} = e^{\tilde{h}_{jk}^i}, \tag{4.30}$$

que seguirá siendo positivo independientemente del valor de $\tilde{h}_{jk}^i$ después de ser actualizado con el algoritmo GD.

4.3.1.2. QuickProp

El principal inconveniente del algoritmo GD es su lenta tasa de convergencia y su alta sensibilidad a la tasa de aprendizaje. La única forma de solucionar estos problemas es introduciendo derivadas de mas alto orden .Los algoritmos GD de segundo orden, como el método de Newton, utilizan explícitamente derivadas de segundo orden. Sin embargo, este método esta limitado en la práctica debido a que el cálculo de la matriz Hessiana $\nabla^2 F_{MMI}$ requiere gran cantidad de cómputo. Por esta razón, se necesita realizar aproximaciones, y ahí es donde entra el algoritmo QuickProp. El algoritmo QuickProp, propuesto originariamente por Fahlman, es una aproximación a los métodos GD de segundo orden y ha sido usado extensamente en sistemas basados en redes neurales.

La derivada de segundo orden con respecto a cualquier parámetro Φde cualquier modelo de idioma en la iteración t, se aproxima por:

$$\frac{\partial^2 F_{MMI}}{\partial \Phi^2}(t) \approx \frac{\frac{\partial F_{MMI}}{\partial \Phi}(t) - \frac{\partial F_{MMI}}{\partial \Phi}(t-1)}{\Phi(t) - \Phi(t-1)}, \tag{4.31}$$

donde $(t-1)$ denota el parámetro de la anterior iteración. En la siguiente iteración los parámetros están estimados por:

$$\Phi(t+1) = \Phi(t) + \Delta\Phi(t), \tag{4.32}$$

esto es, el parámetro actual mas un paso $\Delta\Phi(t)$. El algoritmo QuickProp utiliza una heurística para determinar si el valor aproximado de la matriz Hessiana puede ser usado para actualizar los parámetros. La heurística se resume en los siguientes puntos, en los cuales la ecuación $\Phi(t+1) = \Phi(t) + \Delta\Phi(t)$ se particulariza $\Delta\Phi(t)$ de la siguiente forma:

1. Si el valor actual del gradiente tiene signo opuesto al del gradiente de la iteración anterior, el Hessiano se usa para calcular el paso de una forma análoga al método de Newton :

$$\Delta\Phi(t) = \left[\frac{\partial^2 F_{MMI}}{\partial \Phi^2}(t)\right]^{-1} \frac{\partial F_{MMI}}{\partial \Phi}(t) \tag{4.33}$$

2. Si el valor actual y previo del gradiente tienen el mismo signo, es necesario añadir una constante a ε veces el gradiente actual, de modo que se asegure que el Hessiano sea aplicable. En este caso, el paso resulta ser :

$$\Delta\Phi(t) = \left[\frac{\partial^2 F_{MMI}}{\partial \Phi^2}(t)\right]^{-1} \frac{\partial F_{MMI}}{\partial \Phi}(t) + \varepsilon \frac{\partial F_{MMI}}{\partial \Phi}(t) \tag{4.34}$$

3. Debido a que el calculo del Hessiano es aproximado, no existe garantía de que su valor sea válido y por esa razón el paso podría resultar en un valor demasiado grande. Para tratar con esta situación, el algoritmo QuickProp limita el valor del paso a φ veces el paso de la iteración previa, esto es, $\Delta\Phi(t) = \varphi\Delta\Phi(t-1)$.

4. Después de calcular el paso, si este tiene signo opuesto al valor del gradiente actual, esto implicará que la búsqueda se esta haciendo en la dirección incorrecta. En este caso, el método QuickProp remplaza el Hessiano por cero y solamente el algoritmo GD es usado:

$$\Delta\Phi(t) = \varepsilon \frac{\partial F_{MMI}}{\partial \Phi}(t) \tag{4.35}$$

4.3.1.3. Baum-Welch Extendido

Aunque los algoritmos numéricos anteriores han sido los únicos disponibles para maximizar la función objetivo F_{MMI}, en años recientes se ha encontrado una solución matemática cerrada basada en la del algoritmo Baum-Welch (BW extendido) que permite maximizar la susodicha función objetivo en menos iteraciones, y, por consiguiente, con menos carga computacional y tiempo de entrenamiento. Aunque ya se han descrito las ecuaciones del algoritmo Baum-Welch extendido [26][28][27] , se proporciona a continuación el pseudocódigo de la implementación real del algoritmo Baum-Welch Extendido.

for iter = 1 *to N*
//Calculo de estadisticas
for r = 1 to R
$\lambda_i = \lambda^r$
T_r = #vectores de caractisticas del segmento O_r
for t = 1 : T_r
//Calcula :

$$p(o_{rt}|\lambda_i) = \sum_{j=1}^{M} \frac{w_{ij}}{\sqrt{(2\pi)^D \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}$$

for j = 1 : M

$$\gamma_{rij} = \frac{\frac{w_{ij}}{\sqrt{(2\pi)^D \prod_{k=1}^{D} \sigma_{ijk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{ijk}\right)^2}{2\sigma_{ijk}^2}}}{p(o_{rt}|\lambda_i)}$$

for k = 1 : D
$\theta_{ijk}(o) = \theta_{ijk}(o) + \gamma_{rij} o_{rt}(k)$
$\theta_{ijk}(o^2) = \theta_{ijk}(o) + \gamma_{rij} o_{rt}^2(k)$
end
$\gamma_{ij} = \gamma_{ij} + \gamma_{rij}$
end
$f_{ML}(iter) = f_{ML}(iter) + \log p(o_{rt}|\lambda_i)$
end
//Actualizacin de parametros
for i = 1 to L
for j = 1 to M
$w_{ij} = \frac{\gamma_{ij}}{\sum_{j'=1}^{M} \gamma_{ij'}}$
for k = 1 : D
$\mu_{ijk}^2 = \frac{\theta_{ijk}(o)}{\gamma_{ij}}$
$\sigma_{ijk}^2 = \frac{\theta_{ijk}(o^2)}{\gamma_{ij}} - \mu_{ijk}^2$
end
end
end
end
end

for iter = 1 *to N*
//Calculo de estadisticas
for r = 1 to R
$\lambda_i = \lambda^r$
T_r = #vectores de caractisticas del segmento O_r
for t = 1 to T_r
for i′ = 1 to L

//Calcula y almacena :

$$p(o_{rt}|\lambda_{i'}) = \sum_{j=1}^{M} \frac{w_{i'j}}{\sqrt{2\pi \prod_{k=1}^{D} \sigma_{i'jk}^2}} e^{\sum_{k=1}^{D} -\frac{\left(o_{rtk}-\mu_{i'jk}\right)^2}{2\sigma_{i'jk}^2}}$$

for j = 1 : M
$\gamma_{ri'j} = \frac{p(o_{rt}|\lambda_{i'}^m)}{p(o_{rt}|\lambda_{i'})}$
if(i′ = i) then
$\gamma_{ri'j}^{num'} = \gamma_{ri'j}$
for k = 1 : D
$\theta_{i'jk}^{num}(o) = \theta_{i'jk}^{num}(o) + \gamma_{ri'j}^{num'} o_{rt}(k)$
$\theta_{i'jk}^{num}(o^2) = \theta_{i'jk}^{num}(o) + \gamma_{ri'j}^{num'} o_{rt}^2(k)$
end

$\gamma_{i'j}^{num} = \gamma_{i'j}^{num} + \gamma_{ri'j}^{num'}$
end if
$\gamma_{ri'j}^{den'} = \gamma_{ri'j} \frac{\prod_{t=1}^{T_r} p(o_{rt}|\lambda_r)^{K_r}}{\prod_{t=1}^{T_r} \sum_{q=1}^{L} p(o_{rt}|\lambda_q)^{K_r}}$
for k = 1 : D
$\theta_{i'jk}^{den}(o) = \theta_{i'jk}^{den}(o) + \gamma_{ri'j}^{den'} o_{rt}(k)$
$\theta_{i'jk}^{den}(o^2) = \theta_{i'jk}^{den}(o) + \gamma_{ri'j}^{den'} o_{rt}^2(k)$
end
$\gamma_{i'j}^{den} = \gamma_{i'j}^{den} + \gamma_{ri'j}^{den'}$
end
end
$f_{MMI}(iter) = f_{MMI}(iter) + \log p(o_{rt}|\lambda_i) -$
$- \log \sum_{i'=1}^{L} p(o_{rt}|\lambda_{i'})$

for i = 1 to L
for j = 1 to M
$\tilde{w}_{i'j} = \frac{\gamma^{num}_{i'j} - \gamma^{den}_{i'j} + D_{i'j} w_{i'j}}{\sum_{j'=1}^{M} (\gamma^{num}_{i'j'} - \gamma^{den}_{i'j'}) + D_{i'j}}$
for k = 1 : D
$\tilde{\mu}_{ijk} = \frac{\theta^{num}_{i'jk}(o) - \theta^{den}_{i'jk}(o) + D_{i'j}\mu_{ijk}}{\gamma^{num}_{i'j} - \gamma^{den}_{i'j} + D_{i'j}}$
$\tilde{\sigma}^2_{ijk} = \frac{\theta^{num}_{i'jk}(o^2) - \theta^{den}_{i'jk}(o^2) + D_{i'j}(\sigma^2_{ijk} + \mu^2_{ijk})}{\gamma^{num}_{i'j} - \gamma^{den}_{i'j} + D_{i'j}} - \tilde{\mu}_{ijk}$
$D_{i'j}$ = máx(dos veces el valor que asegure que las varianzas $\tilde{\sigma}^2_{i'jk}$ sean positivas, $\gamma^{den}_{i'j} E$)
//El valor de la constante E suele ser 1 2.
end
end
end
end

El valor del factor $2D_{i'j}$ (dos veces el valor que asegure que las varianzas son positivas) se obtuvo resolviendo la ecuación de segundo grado $\tilde{\sigma}^2_{ijk} = 0$.

4.3.1.4. Comparativa de algoritmos

Se implementaron los algoritmos numéricos de optimización de la función objetivo MMI, Gradient Descent y QuickProp para un conjunto reducido de audio de la evaluación de idioma de NIST 2003 [33](12 idiomas en total). La implementación se realizó transcribiendo directamente las formulas en MATLAB.

En la siguiente figura se resumen los resultados obtenidos de la aplicación de 40 iteraciones del algoritmo Gradient descent sobre el problema de optimización de la función objetivo MMI. Como era de esperar, usando un valor mayor de la tasa de aprendizajeε, la función objetivo se maximiza más rápidamente. No obstante, si el valor de la tasa de aprendizaje es muy elevado la función no convergerá, y empezara a oscilar. Por el contrario, si el valor de la tasa de aprendizaje es muy pequeño, la función convergerá muy lentamente. La gran sensibilidad al valor de la tasa de aprendizaje hace que el algoritmo Gradient Descent sea difícil de aplicar en la práctica.

A continuación se implementó, de forma análoga al caso anterior, el algoritmo QuickProp. Como se observa viendo la siguiente figura la tasa de convergencia de este último es claramente muy superior a Gradient Descent. De hecho, eligiendo convenientemente los parámetros ε y ϕ supera incluso al algoritmo Baum-Welch-Extendido (este algoritmo es el que se empleó en los experimentos sucesivos, debido a su alta eficiencia computacional y rápida convergencia).

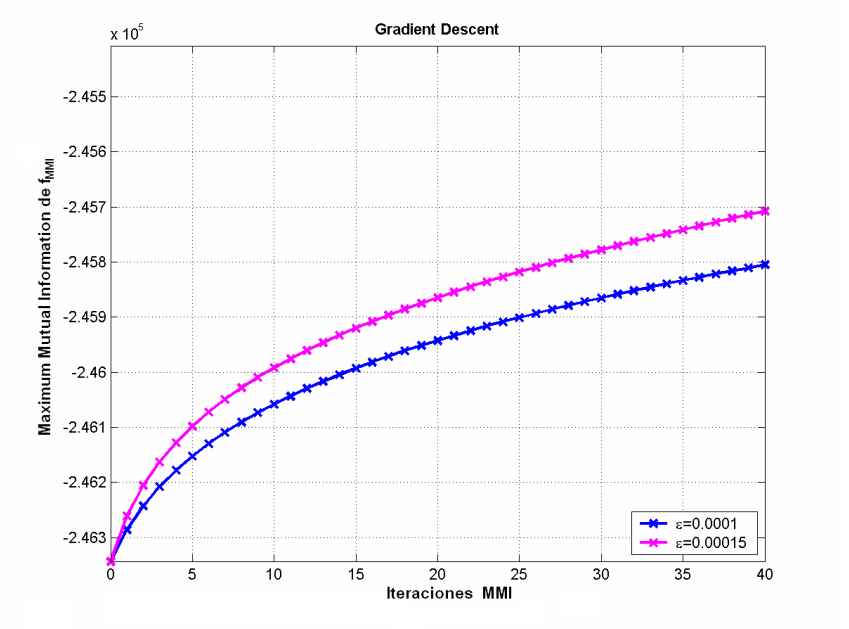

Figura 4.2: Resultados de la maximización de la función objetivo MMI utilizando Gradient Descent sobre el corpus de evaluación de NIST 2003 [33].

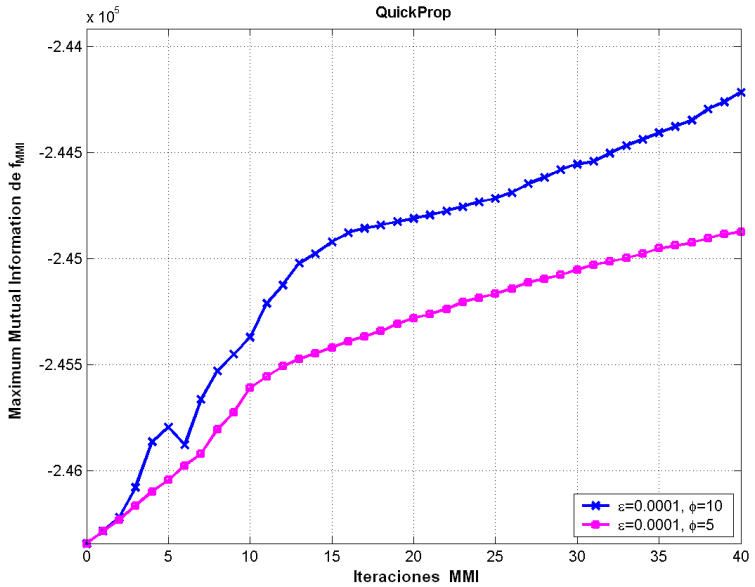

Figura 4.3: Resultados de la maximización de la función objetivo MMI utilizando QuickProp sobre el corpus de evaluación de NIST 2003.

En la siguiente figura se muestran los resultados conjuntos de ambos algoritmos, donde se observa la superioridad de QuickProp:

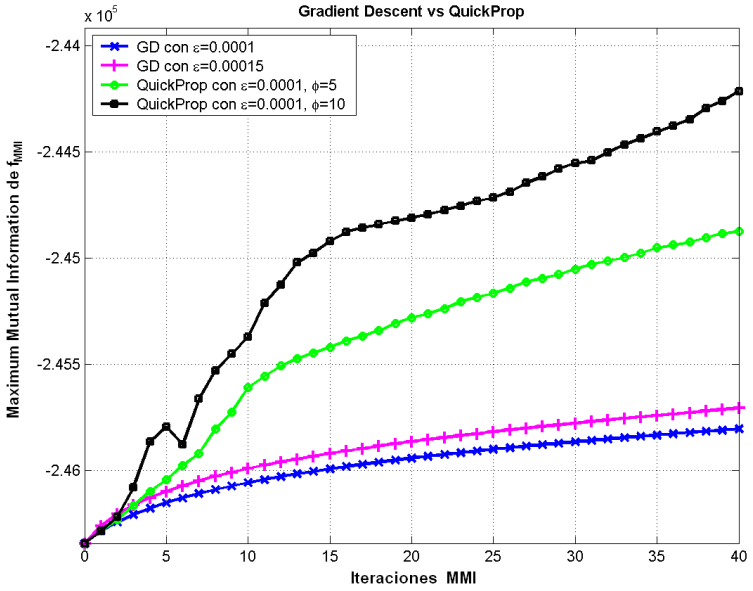

Figura 4.4: Resultados conjuntos de la maximización de la función objetivo MMI utilizando los algoritmos Gradient Descent y QuickProp sobre el corpus de evaluación de NIST 2003.

4.3.2. Ejemplo de aplicación

Se ha diseñado un sencillo experimento en MATLAB para mostrar gráficamente las diferencias entre dos modelos GMM de 2 mezclas entrenadas con dos paradigmas distintos: ML (entrenamiento generativo) y MMI (entrenamiento discriminativo).

Para simplificar el problema solo se consideran dos clases (2 idiomas por ejemplo). Los vectores de características se supondrán bidimensionales para que puedan ser representados en el plano.

Los vectores de características de la clase "1", se compondrán de 10000 puntos distribuidos según una variable aleatoria bidimensional gaussiana con parámetros (media y varianza diagonal) $\mu_{XY} = (0,0); \sigma^2_{XY} = (\frac{1}{2}, \frac{1}{5})$ y 2000 puntos distribuidos del mismo modo pero con parámetros $\mu_{XY} = (2,0); \sigma^2_{XY} = (\frac{1}{2}, \frac{1}{5})$. De forma análoga, los vectores de características de la clase "2" se compondrán nuevamente de 10000 puntos distribuidos según una variable aleatoria bidimensional gaussiana con parámetros $\mu_{XY} = (0,0); \sigma^2_{XY} = (\frac{1}{2}, \frac{1}{5})$ y 2000 puntos distribuidos igualmente con parámetros $\mu_{XY} = (-2,0); \sigma^2_{XY} = (\frac{1}{2}, \frac{1}{5})$. Como se puede observar ambas clases comparten los 10000 puntos y difieren solo en los restantes 2000. La elección de que haya 5 veces más puntos iguales que distintos no se ha hecho al azar. En efecto, en casos prácticos los vectores de características de los idiomas estarán altamente solapados y no serán sino un pequeño número de vectores lo que diferencie un idioma del resto.

En las siguientes figuras se hayan representados los vectores de características (que son simplemente puntos en el plano) para las clases "1" (en color azul) y "2" (en color rojo):

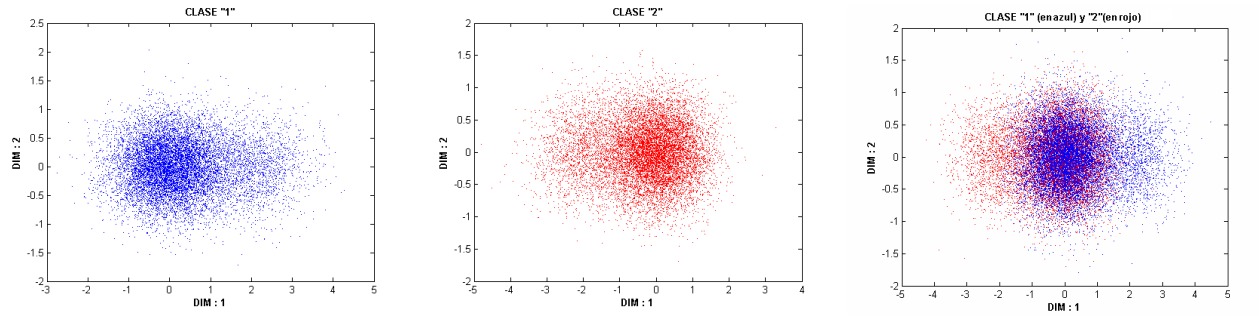

Figura 4.5: Datos de entrenamiento para las clases “1” y “2”.

Una vez generados los vectores de características se procede al entrenamiento de los dos GMM’s. El primero utilizará el criterio ML, y el segundo el criterio MMI. Debido a la simetría de los datos de entrenamiento solo se considerarán los de la clase “2”.

Para el modelo generativo GMM-ML se han realizado 6 iteraciones utilizando el algoritmo Baum-Welch, y para el modelo discriminativo GMM-MMI, 20 iteraciones utilizando el algoritmo Baum-Welch-Extendido, usando todos los puntos de cada clase como un único segmento (por consiguiente solo se tienen dos segmentos de entrenamiento). En el criterio ML, donde el entrenamiento se efectúa a nivel de frame, esto último es irrelevante.

En las siguientes figuras se muestran los resultados del entrenamiento por los dos criterios, donde se representan las curvas de nivel de densidad de probabilidad.

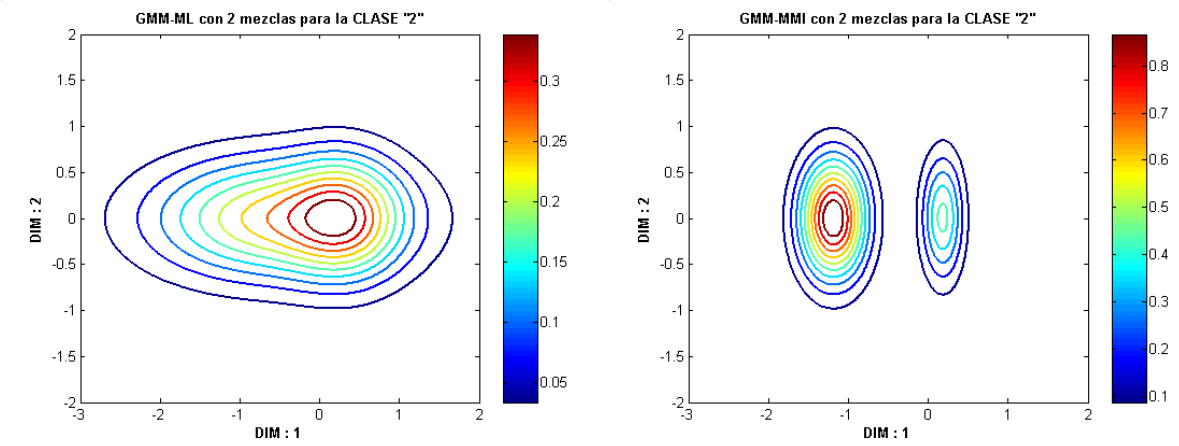

Figura 4.6: Curvas de nivel de densidad de probabilidad para el GMM-ML y el GMM-MMI de la clase 2.

Los resultados del modelo GMM-ML no nos sorprenden en lo más mínimo. En efecto, al ser el modelo GMM-ML generativo los datos de entrenamiento son modelados directamente. Cuanto mayor sea la densidad de puntos de entrenamiento en una región determinada tanto mayor será la probabilidad del modelo GMM-ML en dicha región. De este modo el modelo GMM-ML obtenido es prácticamente idéntico a la función distribución empleada para “generar” los datos de entrenamiento. De ahí el nombre de entrenamiento generativo.

Más interesantes son los resultados del modelo GMM-MMI. Observando las curvas de nivel vemos que, a diferencia del modelo GMM-ML, los datos no son modelados directamente. En otras palabras, la densidad de puntos no se corresponde con la densidad de probabilidad de

las curvas de nivel. De hecho, donde parece haber menos puntos es donde la densidad de probabilidad del modelo MMI es mayor.

La razón de esto es que el criterio MMI modela con mucha mayor probabilidad los puntos que diferencian o discriminan a una clase de la otra. Esta es la característica común de todos los modelos discriminativos, caracterizar las fronteras que separan a las clases.

De esta forma, dado que los vectores de entrenamiento que diferencian a la clase "2" de la clase "1" se encuentran centrados en (X,Y)=(-2,0) el modelo MMI asigna mucha mayor probabilidad a esta región y muy poca a la región que contiene los puntos comunes de ambas clases, y , por otra parte, inútiles para la clasificación.

De esta forma, en las mismas condiciones el criterio MMI obtendrá mejoras siempre que las distribuciones de probabilidad de los datos de evaluación sean parecidas a las de entrenamiento.

Otra consecuencia de todo lo anterior es que debido a que el criterio MMI solo modela los puntos que diferencian a las clases, y puesto que estos suelen ser una minoría en relación al total, no se requieren tantas mezclas para obtener buenos resultados (cuanto mas datos de entrenamiento mas mezclas se deben usar, y viceversa).

En las siguientes figuras se muestran en 3D los modelos de los GMM'S ML Y MMI:

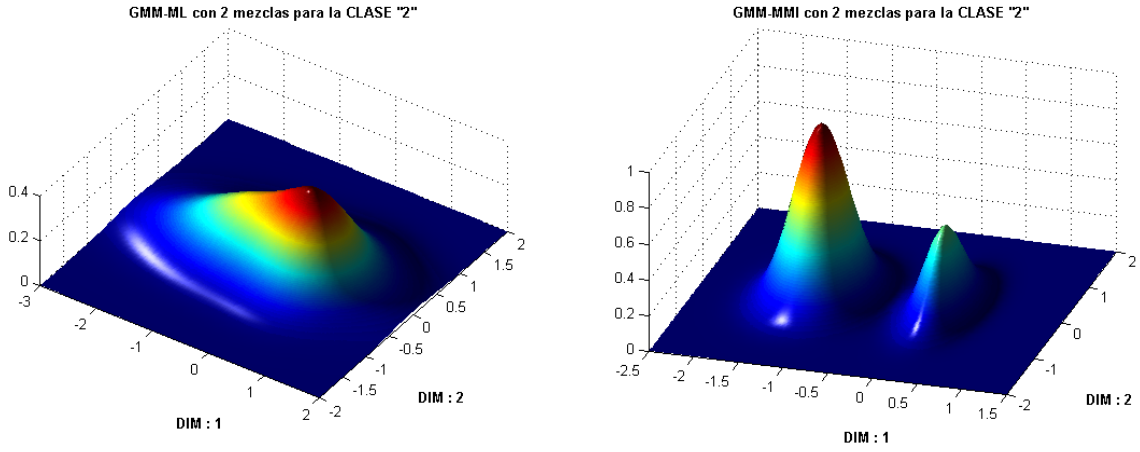

Figura 4.7: Funciones de densidad de probabilidad en 3D para los modelos GMM-ML y GMM-MMI.

Las gráficas de a continuación muestran como se posicionan las mezclas del modelo GMM-ML sobre los datos de entrenamiento de la clase "2", para 2, 8, 32 y 64 mezclas respectivamente. Los puntos rojos representan los centros de las mezclas o gausianas.

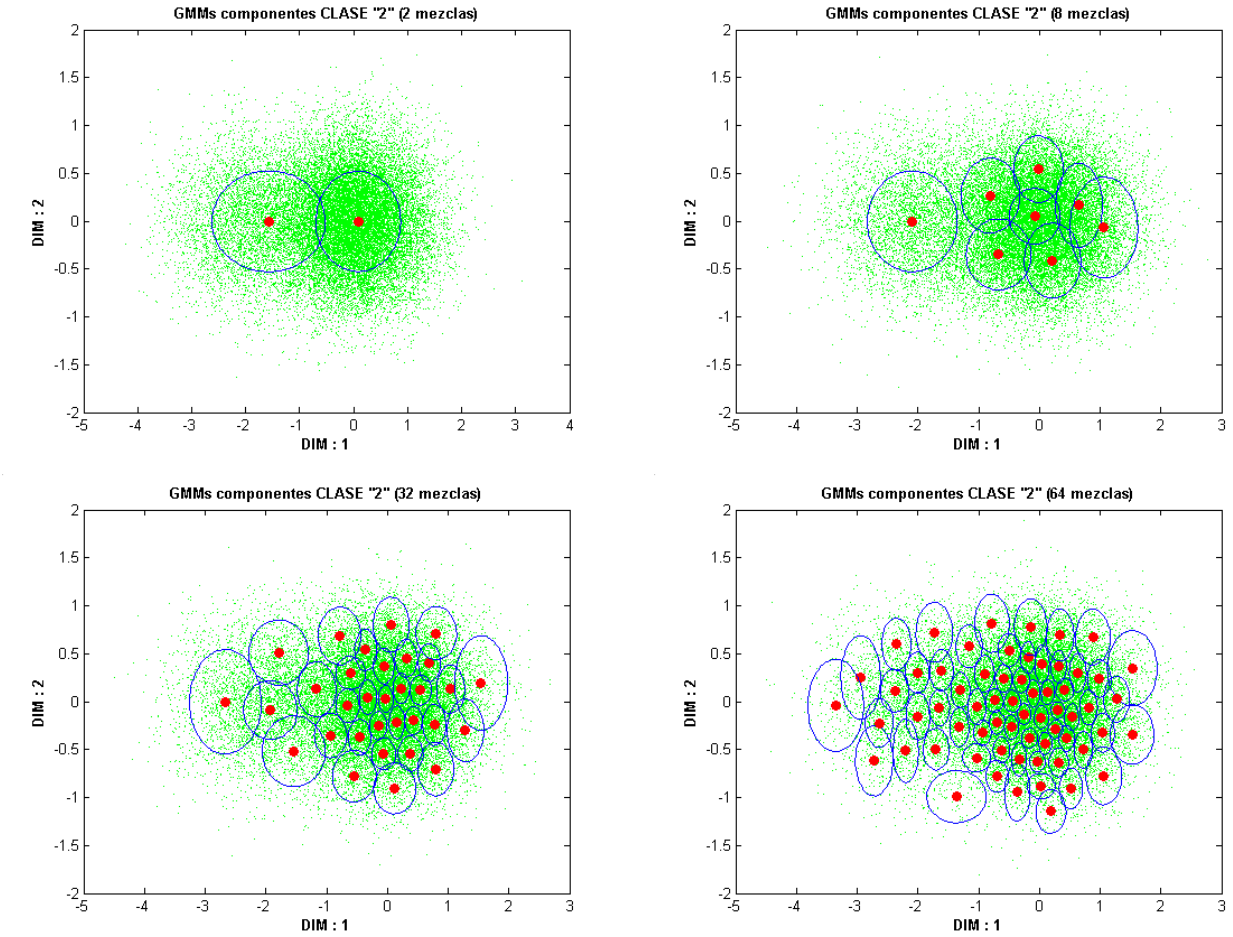

Figura 4.8: GMM's componentes para el modelo GMM-ML.

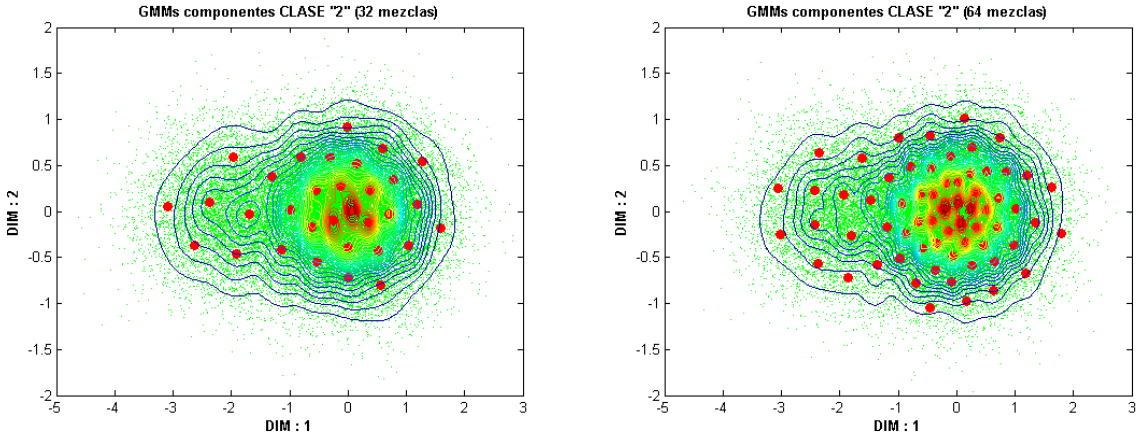

Figura 4.9: GMM's componentes y curvas de nivel para el modelo GMM-ML de 32 y 64 mezclas.

En estas graficas, se aprecia más claramente como la densidad de mezclas es tanto mayor cuanto mayor lo sea la densidad de puntos de la clase "2". Además, vemos como en el modelo GMM-ML de dos mezclas, cada una de ellas se corresponde con mucha precisión con los valores

de varianza y media utilizados para generar los datos de la clase “2”.

5

Diseño y medios

En el siguiente capitulo se mostrara la arquitectura de los sistemas diseñados, junto con los medios (bases de datos, hardware y software) utilizados para su construcción. Se empezará describiendo primeramente las bases de datos utilizadas.

5.1. Medios disponibles

5.1.1. Bases de datos

La base de datos para entrenar los modelos del sistema GMM-MMI proviene del corpus Callfriend. Este corpus, creado por el Linguistic Data Consorcium (LDC) y diseñado específicamente para tareas de reconocimiento de idioma, contiene grabaciones telefónicas en 12 idiomas mas 3 dialectos, a saber : Inglés Americano (dialectos meridional y no meridional), Español Latino (dialecto caribeño y no caribeño), Francés (de Canadá),Árabe (de Egipto), Farsi, Alemán, Japonés, Coreano, Mandarín (dialectos de China y Taiwán). Cada llamada contiene el audio de ambas partes de la conversación telefónica , e información relativa a los locutores (sexo, edad, educación, etc). Para cada idioma existen un total de al menos 80 ficheros de audio de 30 minutos (40 llamadas telefónicas). Para los idiomas con dialectos esta cantidad se duplica. Puesto que ambos locutores comparten el mismo canal telefónico (suponemos que no hablan al mismo tiempo), la cantidad de audio real por llamada será como máximo de 30 minutos. Si a esto le quitamos los silencios producidos por las pausas (tendrán que respirar para continuar hablando), esto nos deja con aproximadamente entre 17 y 19 horas de audio real para cada idioma (el doble para los tres idiomas con dialectos). Debido a la naturaleza informal de las llamadas (por lo general entre conocidos, de ahí el nombre de Call*Friend* el habla se considera espontánea, lo que hace mas difícil la tarea del reconocimiento.

Para la evaluación de idioma de Albayzin, se ha utilizado la base de datos KALAKA, creada por el comité organizador de dicha evaluación. El audio de esta base de datos proviene de la grabación de programas de televisión (informativos, debates, entrevistas, etc) y tiene por lo general un SNR bastante bueno, aunque con condiciones de canal muy diversas (música de fondo, entrevistas en la calle, risas en programas de humor, etc). El audio, originalmente muestreado a 16 Khz, fue necesario convertirlo a 8 KHz (aunque con ello se pierda información, se gana en robustez en el uso de las herramientas de parametrización). En total se dispone de

8 horas de audio para los idiomas de Castellano, Euskera, Gallego y Catalán, que se quedaron en 6 horas después de eliminar los silencios.

5.1.2. Software

Aunque se ha escrito gran cantidad de código para el desarrollo de los sistemas discriminativos MMI y AGMM-SVM (que se estima en mas de 10.000 líneas de código), empleando como lenguajes MATLAB, C++ y script de bash, ha sido inevitable el empleo de herramientas externas de las que se destacan por su importancia las siguientes:

- *HTK*: software desarrollado en la Universidad de Cambridge para problemas de reconocimiento de voz empleando modelos ocultos de Markov. Se ha empleando principalmente para el calculo de los parámetros cepstrales MFCC.
- *HMM Toolkit STK*: software desarrollado por el grupo Speech@Fit de la Universidad Tecnológica de BRNO (República Checa), para el entrenamiento discriminativo MMI y reconocedores fonéticos (Húngaro principalmente). Este software es muy similar en su utilización al paquete HTK (podríamos decir que es una mejora y complemento del anterior).
- *SPHINK-III*: software de reconocimiento de voz desarrollado en la Universidad Carnegie Mellon. Se ha utilizado para el cálculo de los coeficientes cepstrales MFCC.
- *LIVSVM y LIBLINEAR*: software desarrollado en la Universidad Nacional de Taiwán. Con el se han entrenado y evaluado los modelos de maquinas de vectores soporte del sistema discriminativo de contraste AGMM-SVM.

Los parámetros MFCC de HTK se usaron en los primeros experimentos, para terminar utilizando SPHINK-III por motivos de compatibilidad y robustez. Los parámetros MFCC obtenidos por SPHINK-III y HTK son ligeramente distintos, en tanto que emplean distintos bancos de filtros MEL (HTK no escala los filtros para que tengan área unidad, por ejemplo), diferentes transformadas DCT, etc.

5.1.3. Hardware

La mayor parte de los experimentos se han realizado en un portátil con procesador Intel core 2 duo a 1.6 Ghz y 1 GByte de RAM. También se ha empleado el rack de servidores del grupo ATVS, que cuenta con procesadores Opteron de 2.4 Ghz. Gracias al rack se ha conseguido un importante "speedup" en el entrenamiento del sistema MMI, al utilizar varios procesadores en paralelo (hasta 8 al mismo tiempo).

5.2. Diseño

A continuación se describe en detalle los módulos que componen los sistemas discriminativos GMM-MMI-128 y AGMM-SVM-512. El sistema GMM-MMI-128 se ha probado con la evaluación de idioma de NIST 2003, 2005 y la evaluación de ALBAYZIN de 2008. El sistema AGMM-SVM-512 únicamente se ha probado para la evaluación de ALBAYZIN dando excelentes resultados.

5.2.1. Arquitectura de los sistemas MMI y SuperVectors

Por arquitectura entendemos las distintas partes que componen los sistemas de reconocimiento y como se relacionan. Como ya se ha mencionado anteriormente, un sistema de reconocimiento se compone de dos partes claramente diferenciadas: la parte encargada de la extracción de las características del audio (parámetros) que deben ser clasificadas o verificadas y la parte que modela de alguna forma dichas características a través del entrenamiento. A la primera parte también se la conoce como front-end del sistema por ser la primera en realizarse, mientras que a los módulos de entrenamiento y evaluación se les conoce como back-end del sistema por realizarse en última instancia. Describimos a continuación los front-end y back-end de los sistemas MMI y AGMM-SVM desarrollados durante el proyecto.

5.2.1.1. Sistema GMM-MMI-128

5.2.1.1.1. Front-End

Comenzamos describiendo primeramente los front-end's del sistema MMI por constituir el núcleo central del proyecto. El primer front-end del sistema MMI, se desarrolló y probó empleando un conjunto reducido del audio utilizado para la evaluación de idioma de NIST 2003, con el que obtuvo excelentes resultados. Aunque este front-end no se evaluó con rigor en las evaluaciones de idioma, sirvió para sentar las bases de los posteriores sistemas.

Las partes que componen este sistema y sus relaciones, que se pasa a describir a continuación, se muestran en la siguiente figura:

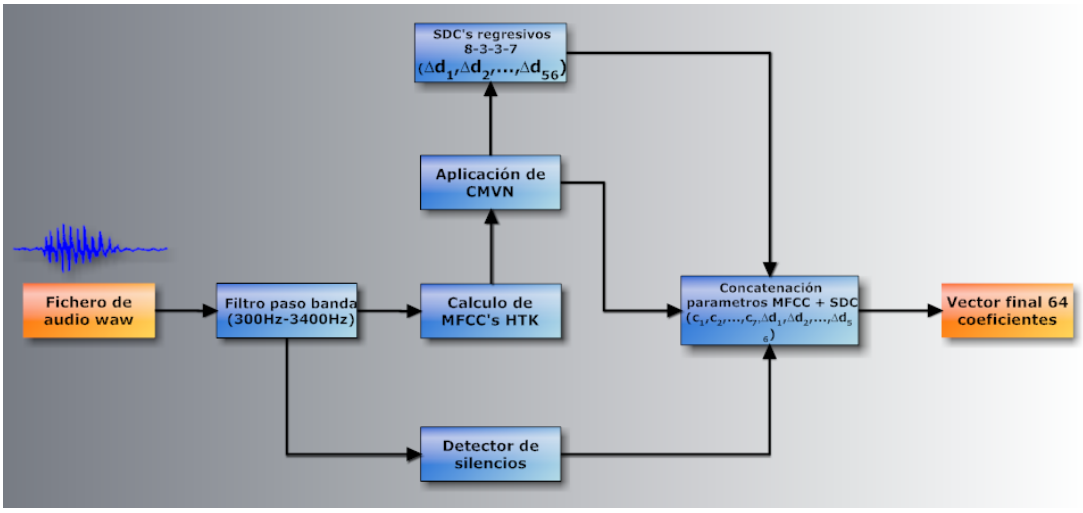

Figura 5.1: Diagrama de obtención de los vectores de parámetros 64-dimensional, empleados durante el desarrollo del sistema GMM-MMI.

Los pasos del front-end para obtener un vector de parámetros de 64 componentes por frame (ventana de 25 ms de audio) son los siguientes:

1. El fichero de audio se pasa por un filtro de butterworth de tercer orden en la banda telefónica de 300 Hz a 3400 Hz (fuera de estas frecuencias no hay información útil por limitaciones del canal telefónico). Al fichero resultante de audio filtrado se le calcula la potencia media espectral frame por frame para detectar los silencios. En la siguiente figura

se muestra el efecto que este filtro tiene sobre el espectrograma del fichero lid00001.wav de la evaluación de idioma de NIST 2003 y su potencia media espectral:

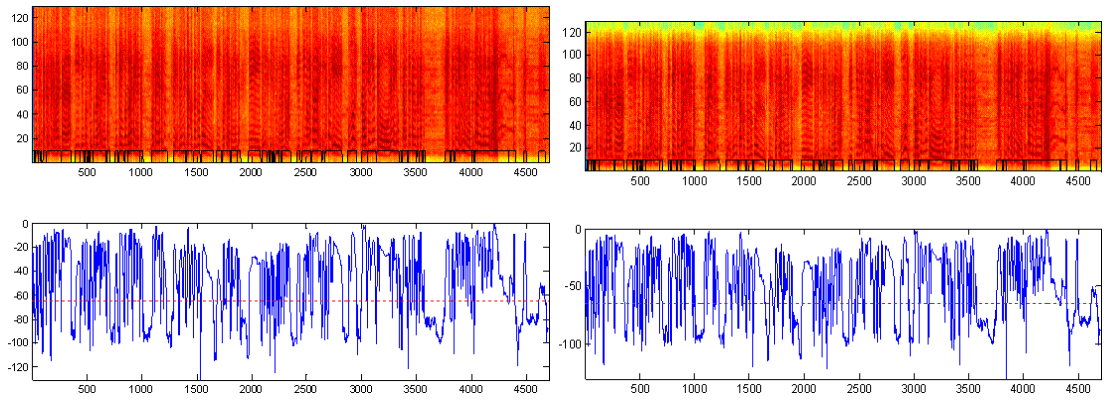

Figura 5.2: Espectrograma del fichero lid00001.wav antes y después del filtrado de Butterworth en la banda de 300-3400 Hz.

2. A continuación, se calcula los primeros 7 coeficientes cepstrales MFCC (sin el coeficiente de energía c_0) del fichero de audio filtrado. Se obtendrán 7 coeficientes MFCC por cada frame del fichero de audio. Para ello se invoca a la herramienta HCOPY de HTK con la siguiente configuración:

   - Coeficiente de pre-énfasis: 0.97.
   - Tamaño de la ventana: 25 ms.
   - Tipo de ventana: Hamming.
   - Tamaño de la FFT: 512 puntos.
   - Frames por segundo (distancia entre enventanados consecutivos): 10 ms.
   - Número de filtros MFCC : 26.

3. A los 7 coeficientes cepstrales MFCC se les aplica CMVN (Cepstral Mean and Variance Normalization).

4. Se calculan los SDC's regresivos a los MFCC compensados del paso anterior para dar lugar a un vector de 56 componentes. El vector resultante se concatena con los 7 coeficientes MFCC dando lugar al vector final de 64 componentes.

5. Basándonos en la información del detector de silencios, eliminamos todos los vectores cuya potencia media espectral del frame correspondiente esté por debajo de -65 dB (decibelios)

Este primer front-end fue desechado al constatarse un rendimiento mediocre cuando se probó con una base de datos mucho más grande como CallFriend (con al menos 50 veces más audio por idioma).

A continuación se muestra el esquema del front-end definitivo usado para las evaluaciones de NIST 2003, NIST 2005 y ALBAYZIN por su mayor rendimiendo que su predecesor:

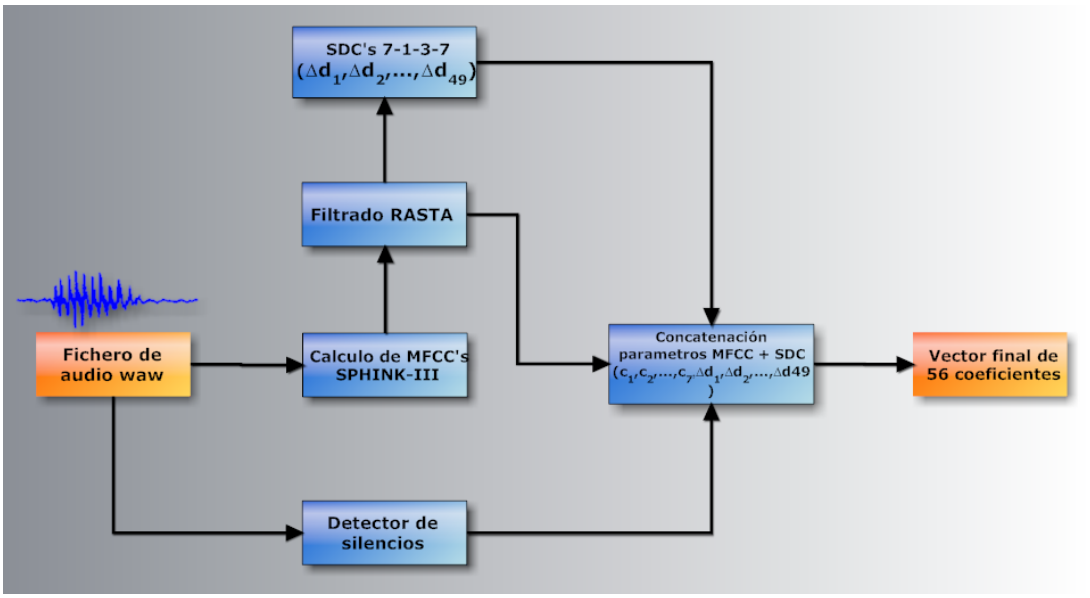

Figura 5.3: Diagrama de obtención del vector de parámetros definitivo (de 56 componentes) del sistema GMM-MMI empleado en NIST 2003, 2005 y ALBAYZIN 2008.

Los pasos para obtener los parámetros cepstrales MFCC se resumen a continuación:

1. El fichero de audio se pasa por un reconocedor fonético de Húngaro (desarrollado por el grupo Speech@Fit) para detectar los silencios. Todo lo que quede en medio de los "fonemas" sil, pau, spk (silencio, pausa, interferencia), y dure mas de 0.5 segundos se considera un segmento valido para el entrenamiento, en caso contrario es descartado.

2. A continuación, se calculan los primeros 7 coeficientes cepstrales MFCC (sin el coeficiente de energía c_0) utilizando SPHINK-III. Se obtendrán 7 coeficientes MFCC por cada frame del fichero de audio. Para ello se invoca a la herramienta wave2feat de SPHINK-III utilizando la siguiente configuración:

   - Coeficiente de pre-énfasis: 0.97.
   - Tamaño de la ventana: 25 ms.
   - Tipo de ventana: Hamming.
   - Frecuencias inferior y superior de los filtros MEL: 130-3700 Hz.
   - Frames por segundo (distancia entre enventanados consecutivos): 10 ms.
   - Número de filtros MFCC: 40.

3. A los 7 coeficientes cepstrales MFCC se les aplica RASTA.

4. Se calculan los parámetros SDC estándar 7-1-3-7 utilizando los MFCC compensados del paso anterior para dar lugar a un vector de 49 componentes. El vector resultante se concatena con los 7 coeficientes MFCC para dar lugar al vector definitivo de 56 componentes.

5. Basándonos en la información del detector de silencios (basado en el reconocedor fonético de húngaro, aunque un detector basado en energía habría funcionado igualmente) eliminamos los vectores cuyos frames has sido descartados por no cumplir los requisitos del paso 1.

5.2.1.1.2. Back-End

En la siguiente figura se muestra el esquema del módulo de entrenamiento del sistema MMI:

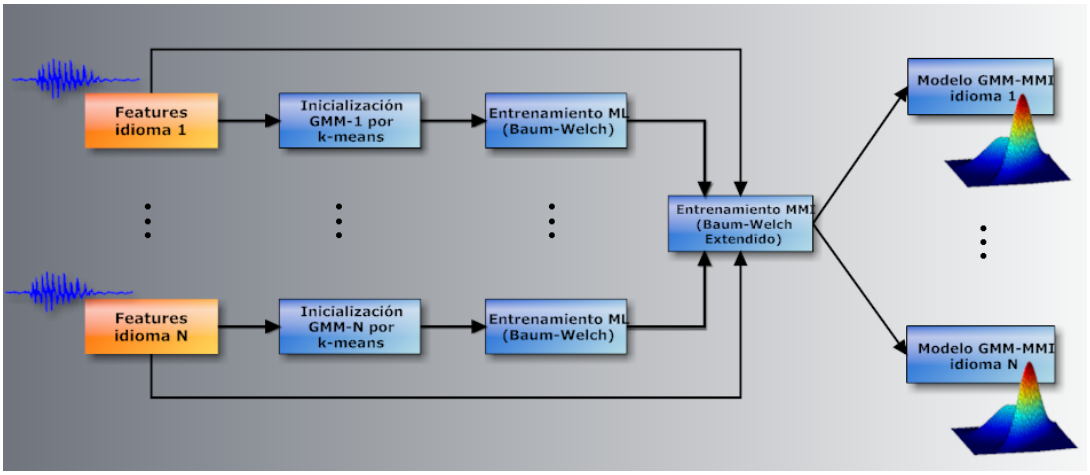

Figura 5.4: Diagrama del modulo de entrenamiento del sistema discriminativo GMM-MMI-128.

Los pasos para obtener los GMM's discriminativos (uno para cada uno de los N idiomas totales) son los siguientes:

1. Utilizando todos los vectores de parámetros disponibles para el idioma i-ésimo, procedemos a inicializar las medias, varianzas y pesos que componen el GMM-i empleando k-means. Las medias se inicializan directamente con los centroides de k-means, las varianzas con la varianza de los vectores (features) que pertenecen a cada centroide respecto de dicho centroide, y los pesos por el número de vectores que pertenecen a dicho centroide dividido por el número de vectores totales (de forma que la suma de los pesos vale la unidad). Se realizan 5 iteraciones de k-means.

2. A continuación se procede a realizar 6 iteraciones de entrenamiento ML por medio del algoritmo Baum-Welch descrito en secciones anteriores.

3. Una vez que se han entrenado todos los GMM's por ML se procede finalmente al entrenamiento MMI, realizando 20 iteraciones del algoritmo Baum-Welch-Extendido, con la constante EBW=2. A diferencia del algoritmo Baum-Welch donde los modelos GMM's son entrenados individualmente empleando únicamente el audio del idioma al que modelan, en cada iteración de MMI se actualizan al mismo tiempo todos los parámetros de los modelos GMM, utilizando para ello todo el audio disponible. Por esta razón, el entrenamiento discriminativo MMI toma N veces mas tiempo de procesado que ML, siendo N el numero de modelos de idioma a entrenar.

Para evaluar el rendimiento del sistema (lo que constituye el segundo componente del back-end), utilizamos como scores las probabilidades a posteriori (en realidad su logaritmo) de que un segmento de test O pertenezca al idioma objetivo s, esto es:

$$score_{GMM-i} = \log P(s|O) = \log \frac{p_{GMM_i}(O|i)^{\frac{1}{T_s}}}{\sum_{k=1}^{N} p_{GMM_k}(O|k)^{\frac{1}{T_s}}}, i = 1, \ldots, N \tag{5.1}$$

La forma de obtener los scores anteriores fue la utilizada en los primeros experimentos sobre un corpus reducido de la evaluación de NIST 2003. En los sucesivos experimentos (lo que incluye la evaluación de idioma de Albayzin 2008) los scores se obtuvieron de realizar la T-Normalización, esto es, obtenidos mediante T-NORM. Hay que indicar, que las probabilidades a priori conocidas $p_{GMM_i}(O|i)^{\frac{1}{T_s}}$ no se pueden utilizar directamente pues solo son comparables a nivel del fichero de test (y por tanto dependiente del fichero).Dicho de otra forma, si el sistema actuara en modo identificación estas probabilidades podrían emplearse directamente (ya que solo se trataría de calcular un máximo), pero resulta que el sistema esta diseñado para operar en modo verificación (lo que requiere que exista un umbral con el que comparar) por requerimientos del sistema de evaluación de NIST LRE.

Los scores T-normalizados se obtuvieron de la siguiente forma:

$$score_{GMM-i}^{T-NORM} = \frac{(\log p_{GMM_i}(O|i)^{\frac{1}{T_s}} - \mu)}{\sigma^2} \quad \text{con } i = 1, \ldots, N \tag{5.2}$$

$$\mu = \frac{1}{N}\sum_{k=1}^{N} \log p_{GMM_k}(O|k)^{\frac{1}{T_s}} \tag{5.3}$$

$$\sigma^2 = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}(\log p_{GMM_k}(O|k)^{\frac{1}{T_s}} - \mu)^2} \tag{5.4}$$

$$\tag{5.5}$$

Una vez obtenidos los scores ya se está en condiciones para obtener las curvas y el EER del sistema. En el siguiente esquema se muestra el módulo de evaluación:

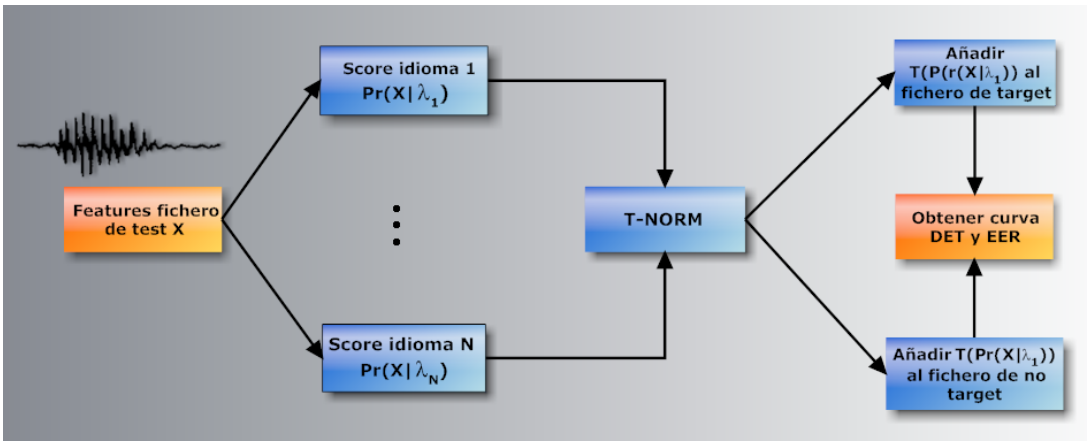

Figura 5.5: Diagrama de evaluación del sistema discriminativo GMM-MMI-128.

5.2.1.2. Sistema hibrido AGMM-SVM-512

El front-end del sistema AGMM-SVM es muy parecido al del sistema MMI. De hecho la única diferencia con respecto a este último esta en que también se hace filtrado RASTA a nivel de parámetros cepstrales después de que CMVN haya sido realizado (en realidad puesto que después de RASTA la media de los parámetros es nula, solo la normalización de la varianza tiene efecto). Aplicando RASTA se obtuvieron mejores resultados.

El audio, originariamente muestreado a 16 KHz, fue convertido a 8 KHz y troceado en segmentos de 30 segundos (si el fichero de entrenamiento original tenia menos de 30 segundos

se dejaba como estaba). En la siguiente figura se esquematiza el proceso de obtención del vector de parámetros:

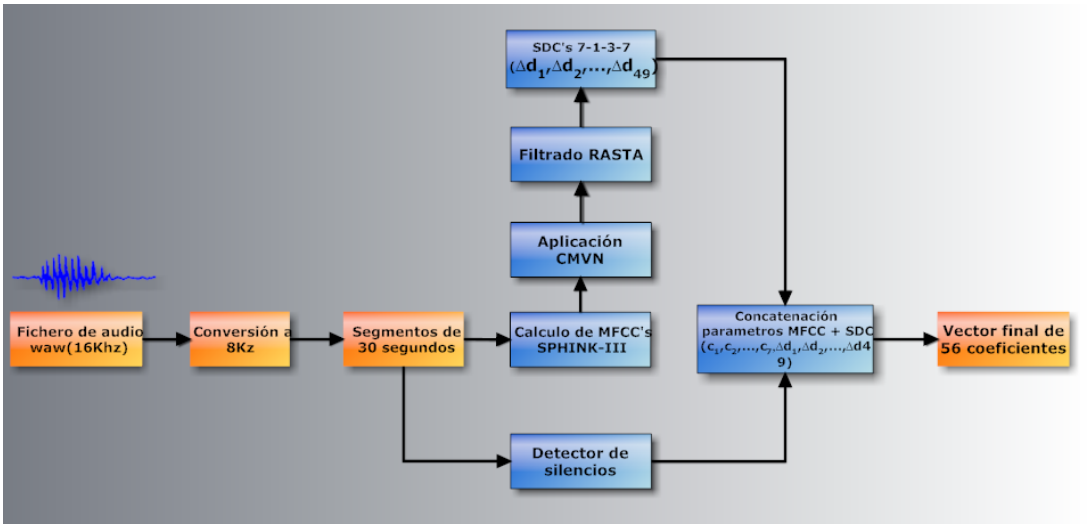

Figura 5.6: Diagrama de obtención del vector de parámetros (de 56 componentes) para el sistema discriminativo AGMM-SVM.

5.2.1.2.1. Back-End

El back-end del sistema AGMM-SVM es en realidad una adaptación del sistema de reconocimiento de locutor que aparece en [35]. A diferencia del sistema GMM-MMI, que emplea los vectores de parámetros directamente, el sistema SVM utiliza SuperVectors para representar de forma compacta las características de un fichero de entrenamiento.

Para obtener los SuperVectors, el back-end del sistema AGMM-SVM lleva a cabo el siguiente procedimiento en cada uno de los idiomas a reconocer:

1. Se entrena un modelo GMM universal (UBM) de 512 mezclas utilizando todo el audio disponible.

2. Por cada fichero de 30 segundos se obtiene un GMM de 512 mezclas por medio de MAP. Solo las medias son adaptadas del UBM con un factor de relevancia de $r = 14$.

3. De cada GMM adaptado (AGMM) se forma un SuperVector concatenando todos los vectores de medias de las 512 mezclas de la siguiente forma :

   SuperVector (j) $= \frac{\mu_{id} - \nu_{id}}{\sigma_{id}} + \nu_{id}$, con j=1,. . . ,512*56 i=1,. . . ,512 d=1,. . . , 56

   donde , μ_{id}y ν_{id}son las medias de la dimensión d-ésima de la mezcla i-ésima del GMM y UBM, respectivamente. σ^2_{id}es la varianza común. El Kernel $\frac{\mu_{id} - \nu_{id}}{\sigma_{id}} + \nu_{id}$ fue el que dio EER mas bajos en la evaluación de Albayzin.

4. Si el SuperVector procede de un fichero de entrenamiento que contiene el idioma objetivo que se esta entrenando, le ponemos la etiqueta +1. De lo contrario le ponemos la etiqueta -1. Añadimos los SuperVectors y sus etiquetas a un fichero te texto plano.

5. Finalmente, utilizando LIBLINEAR (utiliza un Kernel lineal), en concreto su herramienta l-train.exe, y el fichero de texto plano que contiene los SuperVector obtenemos modelo SVM del idioma objetivo. Este proceso se repite en todos los idiomas a reconocer.

En la siguiente figura se resume todo el proceso anterior:

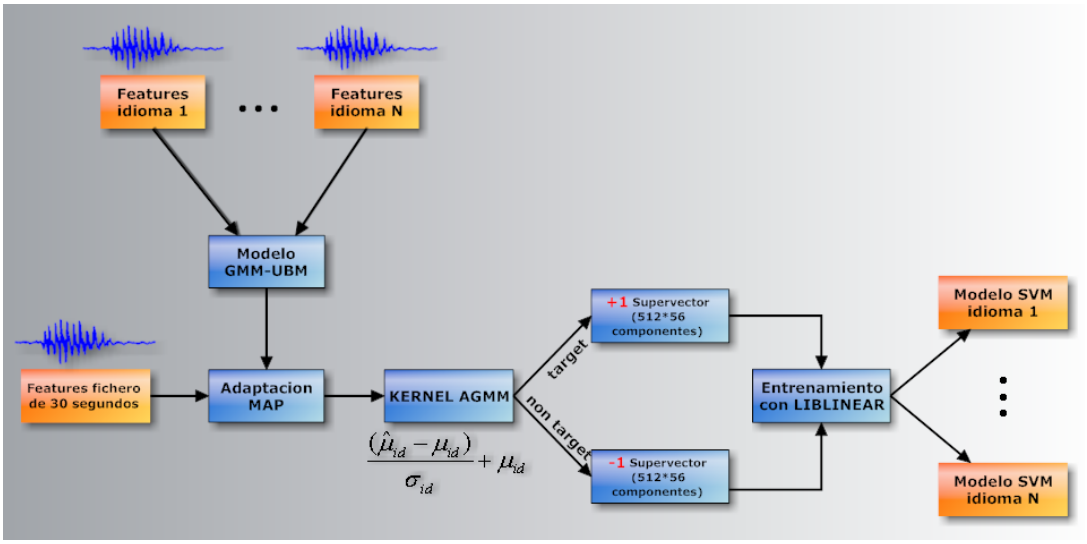

Figura 5.7: Diagrama del modulo de entrenamiento del sistema discriminativo AGMM-SVM-512.

Para la evaluación se realiza un proceso similar al entrenamiento, a saber:

1. Obtenemos el SuperVector del segmento de test O de forma análoga a como se hace durante la fase de entrenamiento. Al SuperVector resultante lo añadimos a un fichero de texto plano (ahora no agregamos la etiqueta pues la desconocemos). Repetimos este proceso para todos los ficheros de test.

2. Utilizando el software l-predict.exe de LIBLINEAR, el fichero con los SuperVector , evaluamos la función que define el hiperplano de separación de todos los SVM, obteniendo los scores $f_{SVM-i}(O)\ \ i = 1, \ldots, N$ (siendo N el número de modelos de idioma). Fue necesario realizar un modificación a la herramienta l-predict para que proporcionara valores decimales en lugar de discretos (esto es, +1 y -1).

3. Realizamos T-Norm con los scores anteriores, de forma similar a como se hizo con el sistema MMI, aplicando la siguientes fórmulas :

$$score_{SVM-i} = \frac{(f_{SVM-i}(O)-\mu)}{\sigma^2} i = 1, \ldots, N \ \text{ con } \mu = \frac{1}{N}\sum_{k=1}^{N} f_{SVM-k}(O)$$
$$\text{y} \qquad \sigma^2 = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}(f_{SVM-k}(O)-\mu)^2}$$

4. Finalmente construimos las curvas DET y calculamos el EER como métrica de rendimiento, utilizando los scores normalizados del paso anterior.

En la siguiente figura se esquematiza todo el proceso anterior:

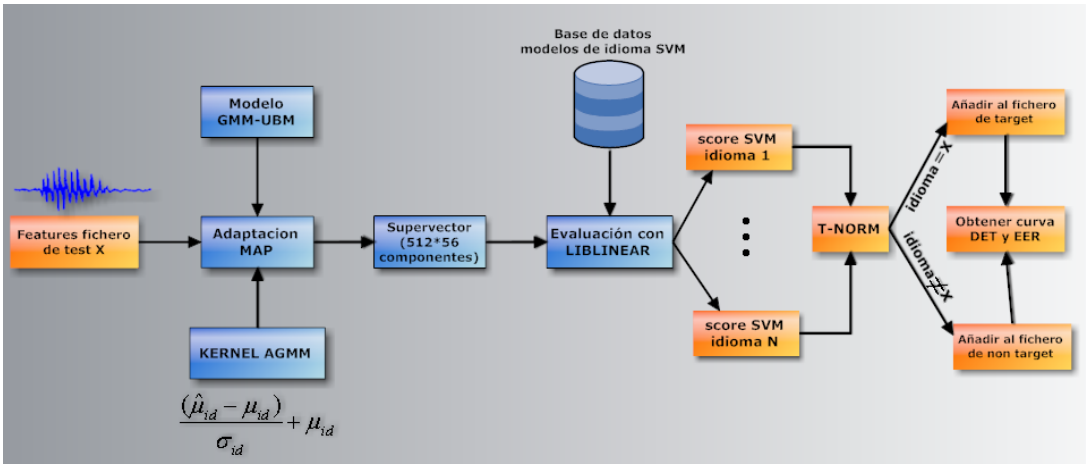

Figura 5.8: Diagrama del módulo de evaluación del sistema discriminativo AGMM-SVM-512.

6

Protocolos, bases de datos y presentación de resultados

En este capitulo veremos las principales métricas empleadas para determinar la calidad de los sistemas de reconocimiento implementados. También veremos los protocolos de evaluación de NIST y Albayzin. Comenzamos describiendo las bases de datos sobre las que se obtendrán dichas métricas de rendimiento.

6.1. Bases de datos

Los comités organizadores de las evaluaciones de verificación de idioma NIST y ALBAYZIN (esta ultima evaluación se ha celebrado por primera vez en 2008) tratadas en este proyecto, son los encargados de seleccionar los ficheros de test a los que se enfrentaran los sistemas. Estos ficheros proceden de porciones no públicas de las bases de datos más importantes para reconocimiento de idioma y que son accesibles para todos los participantes que concurren a la evaluación (NIST entrega a los participantes la base de datos CALLFRIEND para entrenar los modelos, y se reserva una parte de la misma para la evaluación). Las duraciones de los ficheros son de aproximadamente 30, 10 y 3 segundos (en realidad están comprendidas entre 25-35, 7-13, y 2-4 segundos respectivamente) para cada una de las pruebas de verificación.

En las evaluaciones de idioma de NIST LRE '96 y '03, los ficheros de audio de test procedían de la base de datos CALLFRIEND. A partir de la evaluación de idioma de NIST LRE '05, los ficheros proceden en su mayoría de las bases de datos OSHU y MIXER (principalmente de la primera). Los ficheros de test de OSHU tienen una mayor variabilidad que los de CALLFRIEND, haciendo mas difícil su identificación (lo que se constata en la bajada aparente de rendimiento entre las evaluaciones de NIST '03 y NIST '05 , como se verá en los resultados obtenidos por del sistema GMM-MMI en dichas evaluaciones). Todos los ficheros de test de NIST tienen en común que proceden de llamadas telefónicas en líneas de larga distancia (ficheros de audio de 8 Khz, 8 bit/muestra, monocanal, formato $\mu - law$).

Por último, la primera evaluación de ALBAYZIN, celebrada en 2008, emplea una base de datos propia creada ad-hoc para la evaluación. Los ficheros de test proceden de programas de televisión (informativos, debates, entrevistas a pie de calle) y tienen una calidad superior a los empleados por NIST (muestreo a 16 Khz, 16 bits/muestra, monacal). Aunque solo se trata de identificar entre 4 idiomas, el hecho de que dos de ellos sean muy similares (castellano y gallego)

y que exista una gran variabilidad en los ficheros de test (en donde en una misma grabación pueden aparecer varios locutores), sitúan a esta evaluación en un grado de dificultad similar al de las últimas evaluaciones celebradas por NIST ('05 y '07).

6.2. Medidas de rendimiento y presentación de resultados

El último paso en el desarrollo de un sistema de reconocimiento de idioma consiste en evaluar el rendimiento del mismo. Para llevar a cabo esto, al sistema se le presentarán un conjunto de ficheros de test en cada uno de los idiomas para los que fue entrenado. El conjunto de ficheros de test de evaluación, denotado por E, será la unión de N conjuntos disjuntos E_i, esto es,$E = \bigcup_{i=1}^{N} E_i$, cada uno de los cuales contendrá únicamente segmentos de test en uno de los N idiomas posibles. Cada segmento de test se someterá a un total de N pruebas de verificación (véase la figura 2.2). En cada una de estas pruebas de verificación el modelo de idioma i-ésimo se enfrentará a todo el conjunto de evaluación E. El conjunto de ficheros que contengan el idioma del modelo E_i construirán los segmentos de target (usuarios), mientras que los segmentos restantes, esto es $\bigcup_{\substack{j=1 \\ j\neq i}}^{N} E_j$, constituirán los ficheros de test de impostores o non-target. Como resultado de este enfrentamiento se obtendrá una puntuación (score), tanto mas positiva cuanto mayor sea la probabilidad de que el segmento de audio contenga el idioma a verificar. Dicha puntuación se comparará con un valor (umbral de decisión). Si está por encima del umbral se considerará que efectivamente el segmento contiene el idioma i-ésimo, mientras que si está por debajo el segmento será rechazado.

Por otra parte, al porcentaje de ficheros del conjunto E_i que una vez enfrentados al modelo de idioma i-ésimo no superan el umbral (produciéndose por consiguiente un error), se conoce como probabilidad de falso rechazo o pérdida. De modo análogo, al porcentaje de ficheros del conjunto $\bigcup_{\substack{j=1 \\ j\neq i}}^{N} E_j$, que no siendo del idioma i-ésimo se reconocen como tal (produciéndose un error nuevamente), se conoce como probabilidad de falsa aceptación o falsa alarma. Si disminuimos el valor del umbral, aumentaremos la probabilidad de error de falsa aceptación y disminuiremos la probabilidad de error de falso rechazo. Por consiguiente el valor del umbral determinará el punto de trabajo de nuestro sistema. En aplicaciones de seguridad, por ejemplo, donde no queremos que ningún impostor se introduzca en nuestro sistema, elegiremos un valor alto del umbral aunque con ello también aumentemos la probabilidad de error para los usuarios (ficheros de target). Un punto de trabajo interesante para medir la precisión del sistema de verificación, es aquel en el que las probabilidades de error de pérdida y falsa alarma se igualan. El EER (Equal Error Rate) como se le conoce, se usará para medir el rendimiento de los sistemas creados a lo largo del proyecto. Cuanto mas pequeño sea el EER tanto mayor será la precisión del sistema de verificación.

En la siguiente figura, se muestran las curvas típicas de densidad de probabilidad de los scores para los segmentos de target (usuarios) y non-target (impostores). El área de color verde que hay bajo la curva de usuarios representa la probabilidad de falso rechazo, mientras que el área en color azul bajo la curva de impostores representa la probabilidad de falsa aceptación.

Integrando las curvas de densidad anteriores con límites de integración de $[-\infty, umbral]$ y $[umbral, \infty]$ para usuarios e impostores, respectivamente, obtendremos las curvas de distribución de probabilidad, como se muestra en la figura 6.2.

En dicha figura se ha marcado con un punto rojo el EER. Aunque el EER es un buen indicador de la precisión, solo representa uno de los infinitos puntos de trabajo en los que el sistema puede operar. Representado en el eje de las abscisas la probabilidad de falsa alarma y en el eje de ordenadas la probabilidad de pérdida, para todos los valores de umbral posibles,

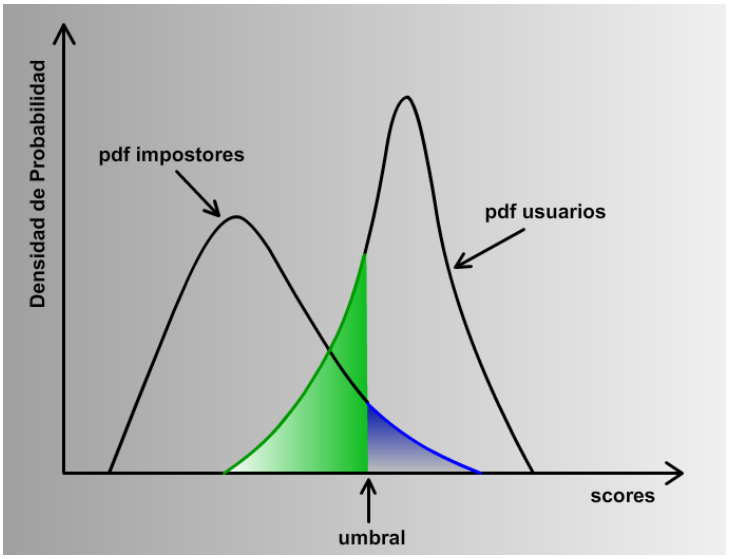

Figura 6.1: Curvas de densidad de probabilidad de las puntuaciones de usuarios e impostores.

obtendremos lo que se conoce como curva ROC (Receiver Operating Characteristic). De esta forma mostramos la precisión del sistema en todos los puntos de trabajo.

Cuando dibujamos en el mismo gráfico varias curvas ROC de distintos sistemas de verificación con rendimiento parecidos, estas tienden a quedar muy próximas, tanto que puede resultar difícil compararlas. Para evitar este problema, Swets [8] y Matin et Al [30] introdujeron una variante de la curva ROC conocida como DET (Detection Error Trade-Off), obtenida realizando una transformación de desviación normal sobre los ejes de la primera. Esta transformación, como demostraremos a continuación, esta motivada por el hecho de que si las distribuciones de probabilidad de impostores y usuarios siguen aproximadamente una distribución normal (condición suficiente, aunque no necesaria) las curvas DET producidas serán líneas rectas (como se verán en las curvas DET resultantes de los numerosos experimentos, por lo general estás tienden a ser líneas prácticamente rectas).

Efectivamente, supuesto que las funciones de densidad de probabilidad de usuarios e impostores sigan una distribución normal con medias μ_1,μ_2, y varianzas σ_1, σ_2 respectivamente:

$$P_M(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} = \phi\left(\frac{t-\mu_2}{\sigma_2}\right) \tag{6.1}$$

$$P_{FA}(t) = \int_{t}^{\infty} \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} = \phi\left(\frac{\mu_1-t}{\sigma_1}\right), \tag{6.2}$$

donde t es el valor del umbral (punto de trabajo) y $\phi(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$ es la función de transformación gaussiana de los ejes.

Si aplicamos $\phi^{-1}(t)$ a $P_M(t)$ y $P_{FA}(t)$, obtendremos, respectivamente:

$$\frac{t-\mu_2}{\sigma_2} = \phi^{-1}(P_M) \tag{6.3}$$

$$\frac{\mu_1-t}{\sigma_1} = \phi^{-1}(P_{FA}) \tag{6.4}$$

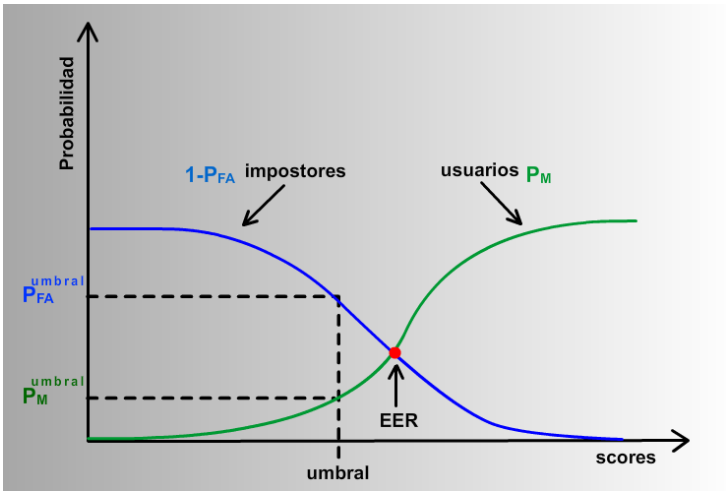

Figura 6.2: Curvas de distribución de probabilidad de las puntuaciones de usuarios e impostores.

Finalmente, despejando en la ecuación (6.4) el valor del umbral t, y sustituyéndolo en (6.3), obtendremos la curva DET:

$$\phi^{-1}(P_M) = -\frac{\sigma_1}{\sigma_2}\phi^{-1}(P_{FA}) + \frac{\mu_1 - \mu_2}{\sigma_2}, \tag{6.5}$$

que como se puede ver corresponde a la ecuación de una recta, quedando así demostrada la linealidad de la curva DET.

A continuación se muestra una curva ROC típica y su curva DET equivalente:

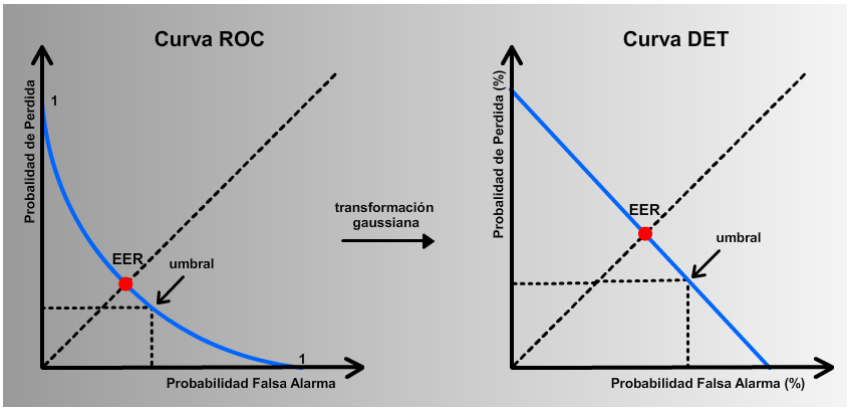

Figura 6.3: Curvas ROC y DET.

6.3. Protocolo de evaluación, evaluaciones NIST LR y ALBAYZIN VL

En esta sección se describe la métrica empleada por NIST LRE (que coincide con ALBAYZIN LV) para determinar el rendimiento y calidad de los sistemas de verificación de idioma que concurren en la evaluación.

Aunque el EER es un valor muy bueno para determinar la calidad de un sistema de verificación, y ha sido, de hecho, la métrica empleada para medir el rendimiento de los sistemas creados en el proyecto, no es este el valor empleado por NIST para la medida del rendimiento. En su lugar se emplea una función de coste (que constituye el núcleo del protocolo de evaluación) promediada para las N pruebas de verificación (supuesto que existan N idiomas objetivo). La función de coste promedio se calcula mediante la siguiente expresión:

$$C_{DET} = \min_{umbral} \left\{ \begin{array}{c} \frac{1}{N}\sum_{i=1}^{N}(C_{perdida}P_{target} \cdot P_{perdida(i)|target(i)}(u)+ \\ +(\sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{(C_{falsa\_alarma}P_{falsa\_alarma(i)|non\_target(j)}(u) \cdot (1-P_{target})}{N-1})) \end{array} \right\} \quad (6.6)$$

El significado de los términos que aparecen en la ecuación anterior es el siguiente:

1. $C_{perdida}$ y C_{falsa_alarma}: costes asociados a las decisiones incorrectas de pérdida y falsa alarma, respectivamente. Tanto NIST como ALBAZYIN les asignan el valor 1 (esto es, ambos tipos de errores se penalizan por igual).

2. P_{target} y $P_{falsa\_alarma} = (1-P_{target})$) : probabilidades a priori de que un fichero de test sea o no, respectivamente, del idioma a verificar. Tanto NIST como ALBAZYIN les asignan el valor 0.5 (esto es, se asume que durante la verificación existe la misma probabilidad de que el fichero sea dr un usuario o de un impostor).

3. $P_{perdida(i)|target(i)}$: probabilidad de error de pérdida para el idioma i-ésimo. Se determinará contando el porcentaje de ficheros de test de idioma i-ésimo que son rechazados, esto es, detectados como de otro idioma (para un umbral determinado).

4. $P_{falsa\_alarma(i)|non\_target(j)}$: probabilidad de error de falsa alarma (o falsa aceptación) para la prueba de verificación del idioma i-ésimo utilizando el idioma j-ésimo como contraste (impostores). Se determinara contando el porcentaje de ficheros del idioma j-ésimo que han sido aceptados erróneamente como pernecientes al idioma i-ésimo a verificar.

Dado que tanto $P_{falsa\_alarma(i)|non\_target(j)}$ como $P_{perdida(i)|target(i)}$ son funciones del valor elegido del umbral u, este se elegirá de forma que la función de coste C_{DET} sea minimizada, y constituyendo este valor el definitivo para la medida del rendimiento.

En la siguientes figuras se muestran los resultados correspondientes a las evaluaciones NIST LR'03 y '05 para todos los equipos participantes (NIST no muestra los nombres de los equipos a los que pertenecen las curvas DET, suponemos que para evitar cualquier tipo de publicidad a los mismos).

Comparando las curvas de resultados de los años 2003 y 2005, se observa paradójicamente una bajada de rendimiento importante de los sistemas de 2005 con respecto a 2003. La razón se debe a que, a partir de 2005, los sistemas se evaluaron con una nueva base de datos (principalmente OSHU) diferente a la tradicional base de datos CALLFRIEND. Esto pone de manifiesto lo dependientes que son los sistemas del audio usado para evaluarlos, aunque los sistemas presentados en 2005 sean claramente superiores y mas complejos que los de 2003.

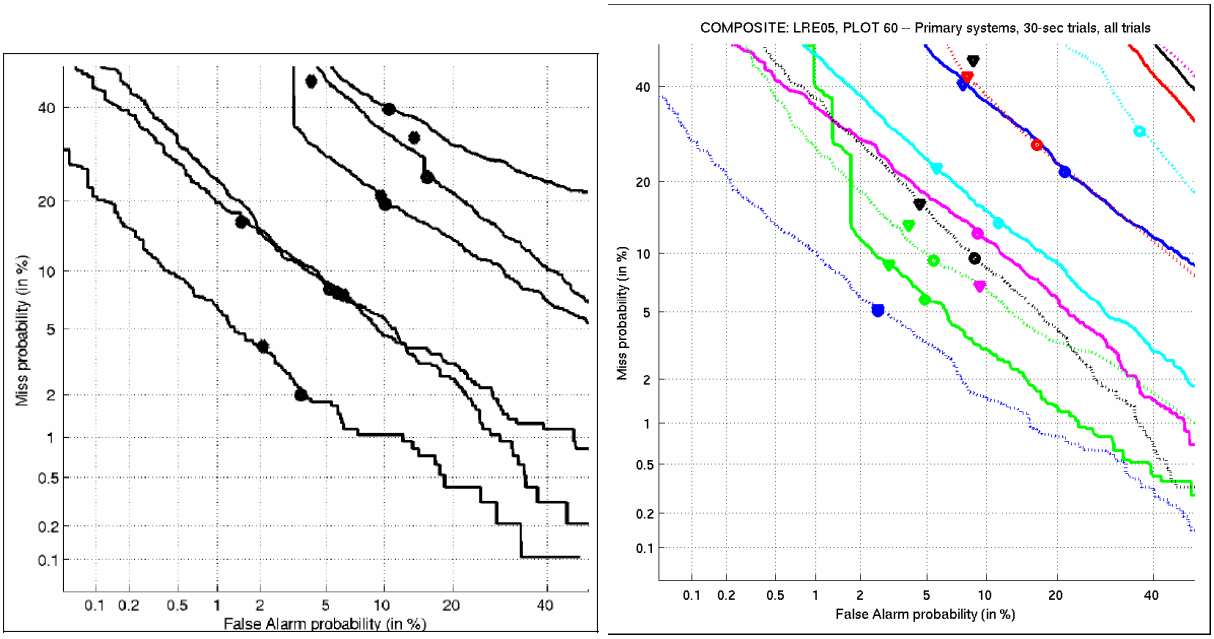

Figura 6.4: Resultados de las evaluaciones LRE NIST 2003 (a la izquierda) y 2005, utilizadas en el proyecto, para la prueba primaria de 30 segundos en conjunto cerrado.

La figura de a continuación muestra los resultados para prueba primaria (segmentos de 30 segundos, en conjunto cerrado y datos de entrenamiento restringidos) de la evaluación de idioma de ALBAYZIN 2008. Como se puede observar solo se presentaron 4 equipos, siendo el ATVS-UAM el sistema ganador con diferencia.

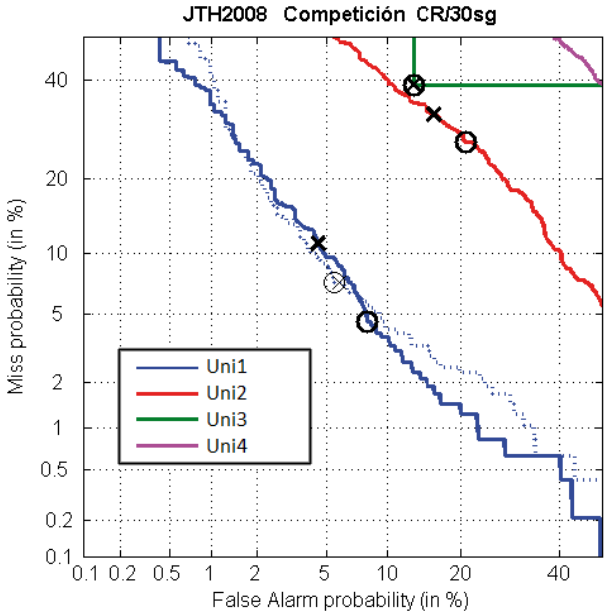

Figura 6.5: Resultados de la evaluación ALBAYZIN08-VL para la prueba primaria.

Posteriormente se compararán los sistemas desarrollados durante el proyecto con respecto a las curvas anteriores.

7

Pruebas y resultados

Hasta el momento solo se ha presentado la parte teórica asociada al problema del reconocimiento de idioma, cubriendo las técnicas que constituyen el estado del arte de los mejores sistemas de reconocimiento (lo que incluye el núcleo del proyecto, el entrenamiento discriminativo MMI, además de técnicas como AGMM-SVM, PPRLM-SVM y Eigenchannel Compensation) y los mas clásicos (GMM y PPRLM). A excepción de los resultados de los algoritmos numéricos de entrenamiento MMI, aún no se han mostrado resultados experimentales prácticos.

En este capitulo se mostrarán finalmente tales resultados aplicados a las evaluaciones de idioma de NIST 2003 [33], 2005 [34] y ALBAYZIN LV 2008 [32]. Aunque se podría haber evaluado también la evaluación de NIST 2007, se prefirió dedicar los esfuerzos computacionales a la evaluación de ALBAYZIN en curso durante el desarrollo del proyecto.

7.1. Experimentos para el desarrollo del sistema GMM-MMI

En esta sección se describen los experimentos realizados para completar con éxito las distintas técnicas que componen el sistema GMM-MMI. Estás técnicas son las que aparecen en las figuras 5.1, 5.3, 5.4 y 5.5. Las dos primeras figuras se corresponden con la obtención de parámetros (front-end) y las dos últimas con los módulos de entrenamiento y evaluación (back-end).

Para estos primeros experimentos de desarrollo, se utilizó un corpus reducido de audio procedente de la evaluación de idioma de NIST 2003, compuesto por 80 ficheros de aproximadamente 30 segundos en los siguientes idiomas: Inglés, Español, Árabe, Farsi (Persa), Francés, Alemán, Japonés, Coreano, Mandarín, Ruso, Tamil y Vietnamita. Esto por supuesto, supone una pérdida de generalidad (al tener menos audio con el que entrenar, la aplicación de técnicas tiene menos impacto en el rendimiento), pero nos permitirá en cambio poder realizar muchos mas experimentos al acortar el tiempo de entrenamiento.

Para completar con éxito el sistema discriminativo primario GMM-MMI del presente PFC (siendo el sistema AGMM-SVM el sistema discriminativo secundario o de contraste) se pasó por diversas etapas con el objetivo de asegurar la correcta implementación de los distintos módulos del mismo. En particular para inicializar los modelos GMM previo al entrenamiento MMI,

primero se inicializan sus medias, varianzas y pesos por medio de K-means y a continuación se entrenan utilizando el criterio ML (resultando lo que en los siguientes experimentos denominaremos el sistema base o GMM-ML). Para comprobar que la inicialización era correcta (esto es, que se había implementado exitósamente el sistema GMM-ML) se implementó el sistema GMM-ML en Matlab y se comparó con el mismo sistema implementado utilizando las funciones del paquete NETLAB [42], ampliamente usado en problemas de reconocimiento de patrones. Posteriormente el código de Matlab se portó a ANSI C para acelerar el entrenamiento (en un orden de casi 20 veces) tomando la precaución de que se obtuvieran los mismos resultados que con el código MATLAB original.

Una vez implementado correctamente el sistema GMM-ML, el siguiente paso natural era implementar el algoritmo Baum-Welch Extendido (en adelante BWE) utilizando los modelos GMM-ML anteriores como punto de partida. A diferencia del sistema GMM-ML del que existen numeras implementaciones con las que comparar (de las que NETLAB es una de muchas), apenas existen herramientas de software utilizables que implementen el criterio MMI utilizando el algoritmo de optimización BWE.

Por suerte, el grupo de voz Speech@FIT de la Univesidad Tecnologica de Brno (BUT), en la Republica Checa, ha desarrollado un ToolKit, denominado STK [43] , basado en el conocido software HTK [21], que, entre otras técnicas, permite entrenar un modelo HMM (Modelo Oculto de Markov) utilizando el criterio discriminativo MMI por medio del algoritmo BWE. Puesto que un GMM puede ser visto como un HMM de un solo estado, en el que cada estado se modela como una mezcla de gausianas, fue fácil invocar a los ejecutables de la librería STK, para entrenar un HMM como si de un GMM se tratase.

De esta forma, si el algoritmo BWE fue implementado correctamente bastará con comparar los modelos obtenidos por el software STK con la implementación propia del algoritmo BWE ,asegurándonos de emplear los mismos modelos GMM-ML iniciales y parámetros de entrenamiento.

La comparación de los vectores de medias, y varianzas de los modelos generados por ambas implementaciones (los pesos no se modifican) arrojó solo diferencias máximas del 1 %. Estas diferencias vestigiales son debidas a errores numéricos de redondeo y truncamiento durante las operaciones de coma flotante (por ejemplo, aunque la expresión $\frac{x^2}{x}$ es equivalente a x, computacionalmente no lo será debido a la precisión limitada de las variables usadas, y, aunque la diferencia entre x y $\frac{x^2}{x}$ sea muy pequeña, si se emplea en sucesivos cálculos, el error se irá acumulando).

Efectivamente, en la siguiente figura se muestran las curvas DET obtenidas entrenando con el software propio (que se ha denominado MMI-ATVS) con el del ToolKit STK . Para entrenar los modelos se utilizaron los 40 primeros ficheros de audio de la evaluación de NIST 2003 (conjunto trainset en la figura), como se comentó mas arriba, y los 40 restantes para la evaluación (conjunto evalset de la figura). La constante k de la figura, cuya función es escalar las probabilidades de los GMM's, hace que aumente la confusión entre las distintas hipótesis alternativas (esto es, modelos a reconocer), incrementando el rendimiento final del sistema. Comparando las curvas observamos que prácticamente son identicas, confirmando la correcta implementación del criterio MMI por BWE. Las curvas en rojo y azul se obtuvieron evaluando sobre el conjunto trainset, de ahí que obtengan mejores resultados (el propósito de este experimento no era ver el rendimiento real, si no ver la completitud en la implementación del algoritmo MMI).

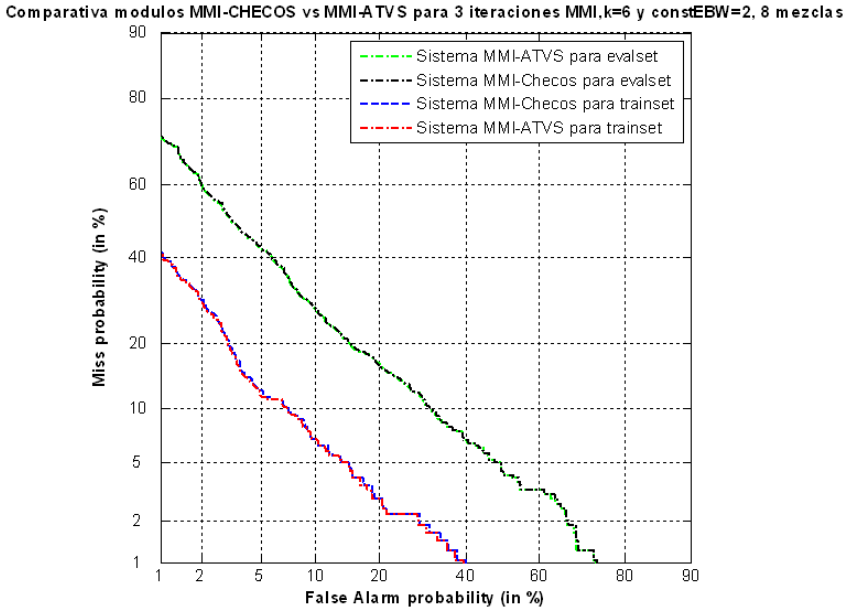

Figura 7.1: Comparativa del sistema GMM-MMI utilizando el software STK del grupo speech@fit y el software implementado.

Implementado correctamente el modulo de entrenamiento del sistema MMI (parte de su back-end), el siguiente paso natural encaminado para incrementar el rendimiento era mejorar la parametrización y el front-end del mismo. Para lograrlo se implementaron las técnicas de compensación de canal clásicas descritas en el capitulo del estado del arte. También se sustituyeron los clásicos coeficientes de velocidad y aceleración obtenidos mediante el software HTK, por los parámetros SDC regresivos y estándar. Estos últimos han probado su superioridad en tareas de reconocimiento de idioma.

7.1.1. Comparación de GMM-ML-MFCC12 con RASTA, CMVN y Feature Warping

A continuación se muestra la mejora como consecuencia de la aplicación de RASTA (**R**el**a**tive **S**pec**tra**), CMVN (**C**epstral **M**ean and **V**ariance **N**ormalization) y Feature Warping sobre los 12 coeficientes cepstrales MFCC (sin el coeficiente cepstral c_0 de enegia) obtenidos mediante HTK.

Los GMM's entrenados por ML, se componen de 64 mezclas, obtenidos y evaluados sobre el corpus de evaluación de NIST 2003 descrito anteriormente. Para aumentar la precisión de los resultados cada fichero de test de los 40 disponibles para cada idioma se evaluó dos veces. Hay que tener en cuenta que debido a la naturaleza aleatoria del proceso de inicialización de los GMM's, incluso entrenando con el mismo audio y condiciones los GMM's generados son ligeramente distintos. La aleatoriedad se introduce al hacer k-means en donde los vectores de medias se eligen aleatoriamente entre todo el audio disponible.

De esta forma la clase target contiene un total de 960 (2*12*40) scores, y la clase de non-target 10560 (2*12*40*11) scores, siendo 12 el número de idiomas de la evaluación de idioma de NIST 2003. Para obtener los scores no se realizo T-NORM, sino que se consideran las probabilidades a posteriori, esto es, $score_{GMM-i} = \log P(s|O)$, como se describió anteriormente.

Estos fueron los resultados obtenidos:

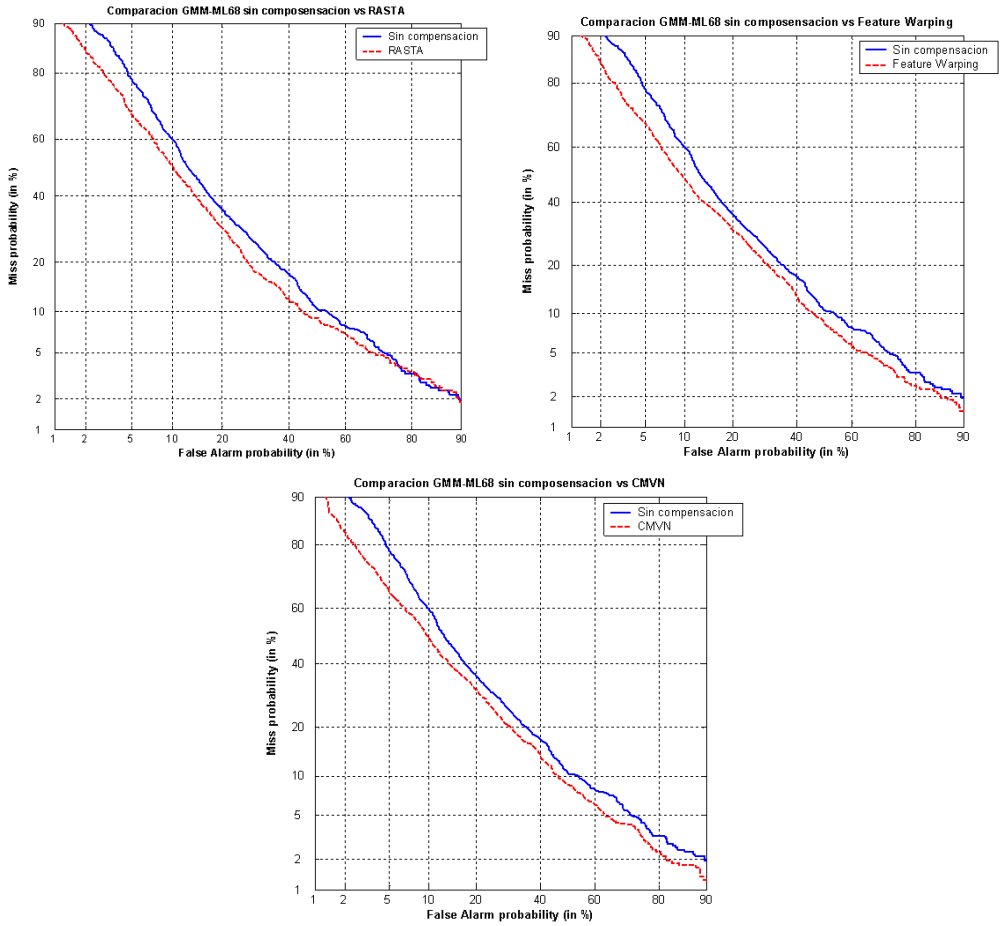

Figura 7.2: Comparativa entre las técnicas clásicas de compensación de canal.

	Sin compensación	CMVN	Warping	RASTA
EER	27.29	24.89	25.00	24.27

Tabla 7.1: Resultados de la aplicación de las técnicas de compensación clásicas sobre el conjunto de prueba de NIST 2003.

De los resultados anteriores se desprende que las tres técnicas parecen reducir por igual el EER. En cualquier caso, las tres técnicas han demostrado ser efectivas para reducir efectos aditivos del canal y de variación lenta. Para feature warping se empleó una ventana de 3 segundos, compuesta por 300 vectores (esto es, a 10 ms por frame).

7.1.2. Comparación GMM-ML-MFCC39 vs GMM-ML-MFCC-SDC64 regresivo

Motivado por los buenos resultados obtenidos en [19] se implementó una versión avanzada de parámetros ceptrales SDC (**S**hifted **D**elta **C**epstral). También se implementó la versión más clásica de parámetros SDC 7-1-3-7 (utilizada por la gran mayoría de sistemas acústicos). En la actualidad, prácticamente el 100 % de los sistemas de reconocimiento de idioma del estado del arte hacen uso de alguna u otra forma de parámetros SDC.

Se comparan los sistemas MFCC39 con el sistema MFFC-SDC64. El sistema MFCC39 esta compuesto de 13 coeficientes cepstrales MFCC (sin c_0), 13 de velocidad y 13 de aceleración. Esto es, cada vector de parámetros se compone de 49 coeficientes (13MFCC + 13Δ + 13$\Delta\Delta$). Los coeficientes deltas de velocidad y aceleración se obtuvieron con HTK. El sistema MFCC-SDC64 esta compuesto de 8 coeficientes MFCC (con c0) y sus correspondientes SDC 8-3-3-7, esto es, 64 coeficientes (8 MFCC + 56 SDC). En la figura 5.1, se describe en detalle el proceso de obtención de este vector de 64 coeficientes (véase la sección 5.2.1.1 , para mas detalles).

Se emplearon GMM de 8 mezclas y se entrenaron por Baum-Welch (criterio ML). Al igual que se hizo con el experimento anterior, los test se repitieron dos veces para aumentar la precisión de los resultados. De este modo el fichero de target contiene 2*12*40 puntuaciones y el de non-target 2*12*40*11 puntuaciones.

Estos fueron los resultados obtenidos:

	MFCC39	MFCC-SDC64
EER	29.58	**23.33**

Tabla 7.2: Resultados del sistema GMM-ML con 64 mezclas, empleando parámetros SDC, sobre el conjunto de evaluación de NIST 2003.

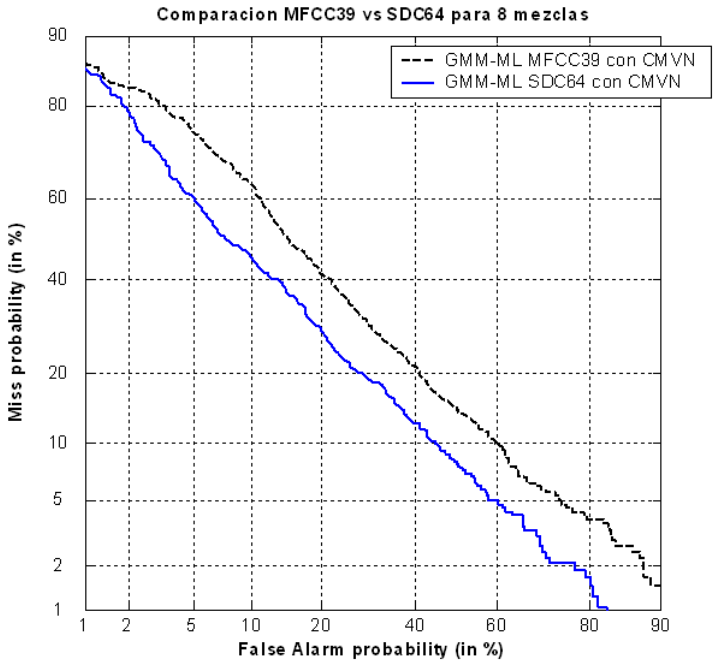

Figura 7.3: Comparación de rendimiento de los parámetros delta clásicos con los coeficientes SDC regresivos.

Como se puede observar el uso de los coeficientes deltas SDC regresivos, tiene un rendimiento considerablemente mayor que el los deltas clásicos. Prácticamente, la totalidad de los sistemas actuales de reconocimiento de idioma, usan los parámetros SDC, haciéndose el uso de este imprescindible para que el rendimiento del sistema sea competitivo. Hay que indicar que en el sistema MMI final que se evaluará con rigor en las evaluaciones de NIST y ALBAYZIN se emplean los parámetros SDC estándar 7-1-3-7.Extrañamente cuando los parametros SDCs regresivos se probaron con bases de datos mas grandes como NIST y ALBAYZIN arrojaron

resultados inferiores a los clasicos SDC's.Se observó tambien que los parámetros SDC estándar no obtenían mejor rendimiento que los delta clásicos a no ser que se usara un número elevado de mezclas, a diferencia de los sdc's regresivos que lo hacían con solo 8 mezclas. Tal vez esto último (y la motivación de los buenos resultados obtenidos en [19]) fue lo que hizo infravalorar el uso de los sdc's clasicos 7-1-3-7 desde el principio.

7.1.3. Comparación GMM-MMI SDC vs GMM-ML SDC en evalset y trainset

Finalmente, mostramos los resultados del sistema GMM-MMI sobre el corpus de audio de prueba. Como veremos, incrementado la información mutua entre los segmentos de audio (en este caso, ficheros de 30 segundos) y los idiomas a los que pertenecen, esto es , optimizando la función objetivo MMI, se logra obtener resultados considerablemente superiores a los obtenidos por el uso de los clásicos GMM's generativos (esto es, entrenados por ML). La función objetivo MMI que nos ocupa, ya descrita anteriormente, aunque no en la forma presentada a continuación, es la siguiente:

$$f_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{P(O_r^{T_r}|\lambda^r)^{\frac{k}{T_r}}}{\sum_{i=1}^{L} P(O_r^{T_r}|\lambda_i)^{\frac{k}{T_r}}}, \tag{7.1}$$

siendo R el número de segmentos de audio totales para el entrenamiento (de todos los idiomas), T_r el numero de vectores del segmento r-ésimo O_r, L el número de idioma, λ^r el modelo GMM de idioma contenido en el segmento de audio O_r (ahora representado por T_r vectores de parámetros MFCC, SDC, etc). Por último la constante k, que originariamente no aparece en la formulación del criterio MMI, se introduce para compensar suposiciones incorrectas en la determinación de la probabilidad $P(O_r^{T_r}|\lambda^r)$, además de incrementar la confusión entre modelos, lo que redunda en una mejora en la discriminación entre idiomas, y, finalmente, en un incremento de la precisión. Eliminando el denominador en $f_{MMI}(\lambda)$ tendremos el criterio ML ya visto con anterioridad, esto es:

$$f_{ML}(\lambda) = \sum_{r=1}^{R} \log P(O_r^{T_r}|\lambda^r)^{\frac{1}{T_r}} = \sum_{r=1}^{R} \frac{1}{T_r} \log P(O_r^{T_r}|\lambda^r), \tag{7.2}$$

donde se ha eliminado el denominador y la constante k. El factor T_r normaliza la probabilidad por el número de frames (de lo contrario no se podría comparar las puntuaciones de ficheros de 3 segundos con los de 30 segundos, por ejemplo).

La figura 7.4 muestra el resultado de optimizar por MMI sobre el conjunto de evaluación de NIST 2003 (se utilizó la primera mitad de los ficheros para entrenamiento, y la segunda para evaluación, de un total de 80 ficheros por idioma). Al igual que en experimentos anteriores, los parámetros se obtuvieron de acuerdo a la figura 5.1. Para el sistema GMM-ML se realizaron 6 iteraciones del algoritmo Baum-Welch (ML converge en aproximadamente 4 iteraciones, se hizieron dos más para asegurarnos). Por ultimó, a los modelos GMM-ML se les realizó otras 6 iteraciones por Baum-Welch-Extendido para obtener finalmente los modelos GMM-MMI.

Como se puede observar viendo dicha figura, el rendimiento del sistema MMI es notablemente superior al sistema GMM-ML de partida, sobre todo cuando el valor de la constante k se ajusta a 6. El valor de k que maximiza el rendimiento de la función MMI debe ser determinado experimentalmente, siendo dependiente de la parametrización empleada y de la cantidad de audio disponible para el entrenamiento. De acuerdo a los experimentos, cuando se dispone de más audio, el valor óptimo de k se incrementa, o indirectamente, un mayor número de mezclas en los modelos, disminuye el valor óptimo de k. Esto último se debe a que se necesitan más mezclas cuando se dispone de más audio para obtener resultados precisos.

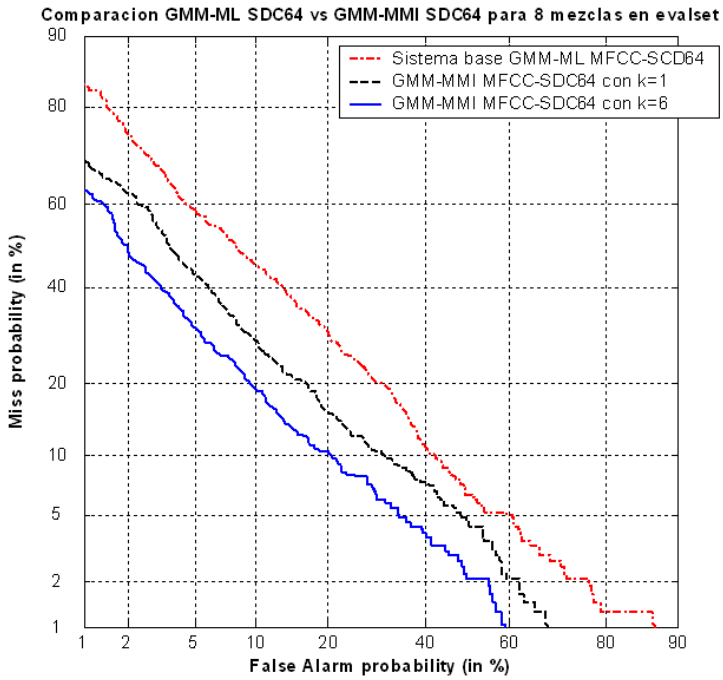

Figura 7.4: Curvas DET para los sistemas GMM MMI y ML sobre el conjunto de evaluación de prueba.

Se repitió el experimento anterior empleado el mismo audio de evaluación que de entrenamiento. Aunque esto en si no tiene ningún valor práctico, nos permite ver la capacidad de aprendizaje del sistema MMI respecto del sistema ML en su máxima expresión. En la figura 7.5 se muestran las curvas DET obtenidas (mismo procedimiento que en el experimento anterior):

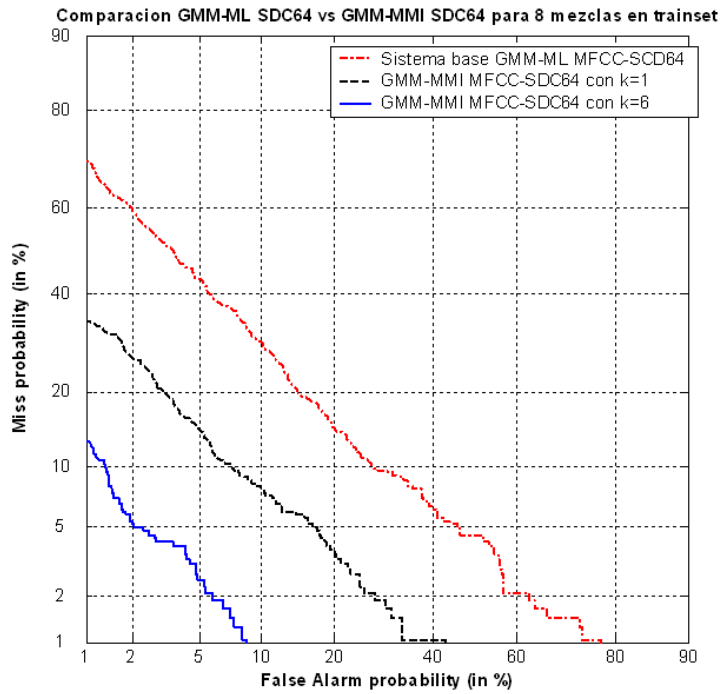

Figura 7.5: Curvas DET para los sistemas GMM MMI y ML sobre el conjunto de entrenamiento de prueba.

Observando la figura 7.5 se desprende que el sistema MMI tiene una capacidad de memorización o aprendizaje del audio de entrenamiento muy superior al sistema ML. Esto hace que el rendimiento también mejore incluso aunque el audio de evaluación no se presente en el de entrenamiento.

Finalmente, los resultados numéricos de ambas pruebas se resumen en la siguiente tabla:

	GMM-ML SDC64	GMM-MMI SDC64 ,k=1	GMM-MMI SDC64 ,k=6
EER evalset	**24.37**	**18.12**	**13.95**
EER trainset	17.50	8.75	**3.95**

Tabla 7.3: Resultados del sistema GMM-ML con 8 mezclas, empleando parámetros SDC, sobre el conjunto de evaluación de NIST 2003.

Podemos ver que el sistema MMI en el conjunto de evaluación prácticamente reduce a la mitad el EER del sistema base de partida. En el conjunto de entrenamiento el EER del sistema GMM-ML se cuadruplica respecto del sistema MMI.

7.1.4. Rendimiento del sistema completo GMM-MMI128-SDC

Se repitió el experimento anterior aumentado el número de mezclas de los GMM's a 128. Utilizar un número de mezclas mayor provoca que los modelos se ajusten mucho al audio de entrenamiento, reduciendo la generalización en el audio de evaluación. Esto es, para el escaso audio disponible (aproximadamente 600 segundos por idioma), 128 mezclas fue el valor óptimo.

A diferencia del entrenamiento ML, en MMI el número de mezclas no tiene un impacto tan importante como en ML y con un número pequeño de mezclas ya se consiguen buenos resultados. Esto se debe, como ya se menciono anteriormente, a que en MMI se modela únicamente la parte del audio de un idioma que difiere respecto del resto, al contrario que en ML, donde se modela la distribución de todo el audio del idioma. Puesto que más audio requiere emplear mayor número de mezclas para ajustar los modelos a la distribución de probabilidad de los vectores de parámetros del audio, esto significa que MMI requiere menor número de mezclas, al tener menos vectores de parámetros que modelar.

En MMI si queremos mejorar el rendimiento deberemos aumentar la generalización lo que se conseguirá ajustando el valor de escala de probabilidades y, principalmente, aumentando la cantidad de audio con que entrenamos los modelos (que para el sistema GMM-MMI128-SDC es muy reducido).

Esta fue la gráfica obtenida:

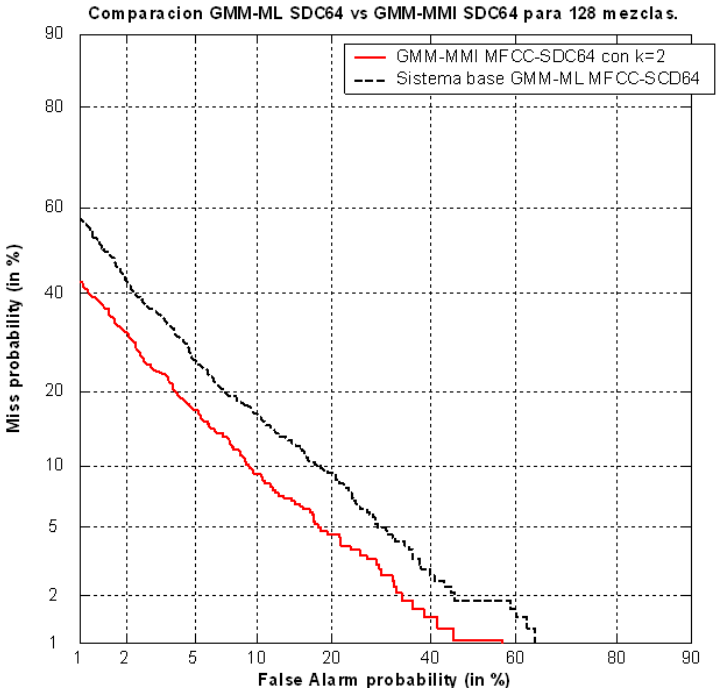

Figura 7.6: Comparación de rendimiento de los parámetros delta clásicos con los coeficientes SDC regresivos.

Y los valores de EER de ambos sistemas:

	GMM-ML SDC	GMM-MMI SDC
EER	**13.54**	**9.58**

Tabla 7.4: Resultados del sistema GMM-ML con 64 mezclas, empleando parámetros SDC, sobre el conjunto de evaluación de NIST 2003.

En vista a los resultados de la tabla anterior, el rendimiento del sistema MMI puede parecer modesto. Sin embargo, hay que tener en cuenta que el EER de 13.54 fue el mejor resultado obtenido para el sistema GMM-ML, y, que el audio de entrenamiento es muy reducido (aproximadamente 10 minutos por idioma). Posteriormente se evaluará NIST 2003 utilizando todo el audio de CALLFRIEND (aproximadamente 20 horas por idioma), donde se verá que el paradigma de entrenamiento MMI es claramente muy superior a ML.

7.1.5. Aplicación de Z-NORM al sistema GMM-MMI

A iniciativa del tutor se diseñó un experimento en donde se realiza un entrenamiento discriminativo mas enfocado a la verificación del idioma que a la identificación.

Hasta ahora, para cada fichero de entrenamiento disponible, los modelos GMM de todos los idiomas se actualizaban en cada iteración, de forma que el modelo de idioma que contenía el fichero, incrementaba la probabilidad de que dicho fichero perteneciera a el, mientras que para el resto de idiomas esta probabilidad disminuía.

Como ya se indicó al principio del proyecto, en un sistema operando en modo verificación

(véase la figura 1.2), para saber si un fichero de test O contiene el idioma objetivo representado por el modelo λ, se debe aplicar la siguiente resta y compararla con un umbral:

$$score = S(O|\lambda) - S(O|\lambda') \tag{7.3}$$

El modelo λ' representa el "anti-idioma" (también conocido como modelo universal) del idioma objetivo, e intenta modelar todo aquel audio que no contenga el idioma objetivo. Si bien el modelo λ está bien definido (ya que solo hay que entrenar un GMM utilizando el audio del idioma objetivo, el cual es bien conocido), el antimodelo solo puede aproximarse realizando diversas asunciones (como por ejemplo suponer que el audio alternativo no es infinito sino que solo contiene los restantes idiomas dentro de una lista predefinida).

La formula utilizada hasta ahora para obtener los scores del sistema MMI puede reescribirse de la siguiente forma:

$$\begin{aligned} Score_{GMM-i} = \log P(s|O) &= \log \frac{p_{GMM_i}(O|i)^{\frac{1}{T_s}}}{\sum_{k=1}^{N} p_{GMM_k}(O|k)^{\frac{1}{T_s}}} = \log p_{GMM_i}(O|i)^{\frac{1}{T_s}} - \log \sum_{k=1}^{N} p_{GMM_k}(O|k)^{\frac{1}{T_s}} \\ i &= 1, \ldots, N \end{aligned} \tag{7.4}$$

que como puede observarse, se ajusta a la de un sistema de verificación, en donde al antimodelo lo aproxima sumando la probabilidad de todos los modelos de idioma. Lo mismo sucede si analizamos la fórmula de un sistema GMM-UBM.

Siguiendo el esquema del sistema de verificación descrito, ahora para cada idioma se obtendrán dos GMM's uno entrenado utilizando el audio exclusivo del idioma objetivo y otro utilizando el audio restante. Los GMM's serán entrenados por medio de los criterios ML y MMI como se hizo para el sistema GMM-MMI-8-SDC descrito en la sección anterior. Después de entrenar los modelos obtendremos los scores de evaluación utilizando:

$$score_i^{MMI}(O) = \log p_{GMM-MMI_i}(O|i)^{\frac{1}{T_s}} - \log p_{GMM-MMI_i}(O|i')^{\frac{1}{T_s}} \quad i = 1, \ldots, N \tag{7.5}$$

$$score_i^{ML}(O) = \log p_{GMM-ML_i}(O|i)^{\frac{1}{T_s}} - \log p_{GMM-ML_i}(O|i')^{\frac{1}{T_s}} \quad i = 1, \ldots, N, \tag{7.6}$$

siendo O el fichero de test, i el modelo de idioma i-ésimo, i' su antimodelo y N el numero de idiomas totales.

Además, como novedad, también se aplicara Z-Norm a las puntuaciones anteriores empleando la siguiente expresión (véase la sección de Z-Norm en el capitulo de compensación de canal):

$$\begin{aligned} score_i^{Z-NORM}(O) &= \frac{(score_i(O) - \mu_i)}{\sigma_i} \\ \mu_i &= \frac{1}{M_i} \sum_{O' \notin i} score_i(O') \\ \sigma_i &= \sqrt{\frac{1}{M_i - 1} \sum_{O' \notin i} (score_i(O') - \mu)^2} \\ i &= 1, \ldots, N \end{aligned} \tag{7.7}$$

A continuación se muestran los resultados de ambos sistemas ML y MMI.

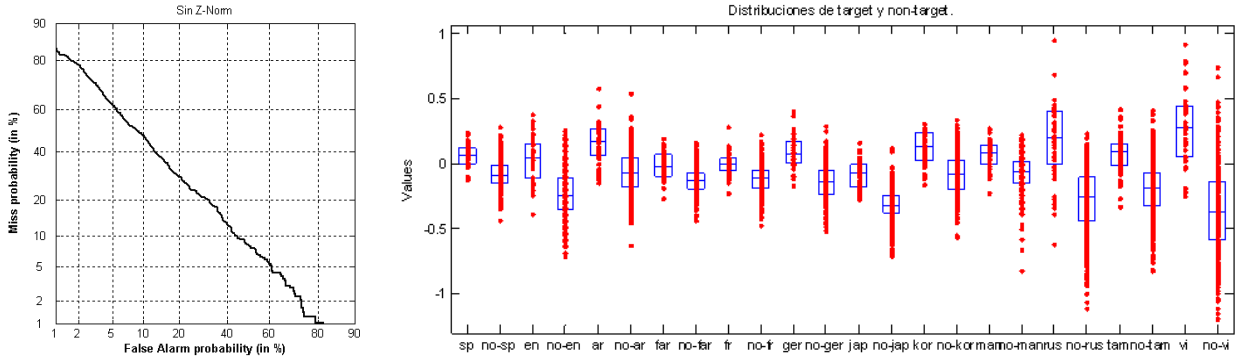

Figura 7.7: Curva DET del sistema GMM-ML sin Z-Norm y grafica de distribución de puntuaciones.

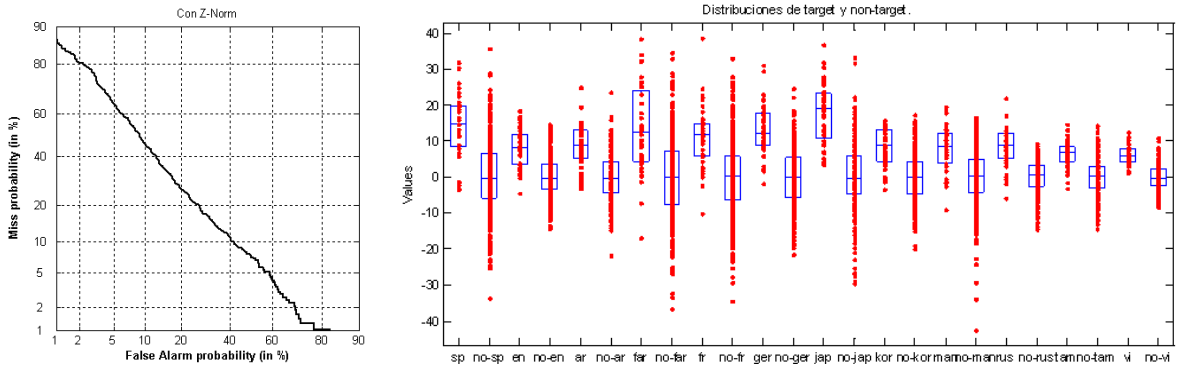

Figura 7.8: Curva DET del sistema GMM-ML con Z-Norm y grafica de distribución de puntuaciones.

	Sistema GMM-ML-SDC-64	Sistema GMM-ML-SDC64 binario sin Z-Norm	Sistema GMM-ML-SDC64 binario con Z-Norm
EER	24.37	24.42	22.57

Tabla 7.5: EER para el sistema GMM-ML-8-SDC antes y después de aplicar Z-NORM, sobre el conjunto de evaluación de NIST 2003.

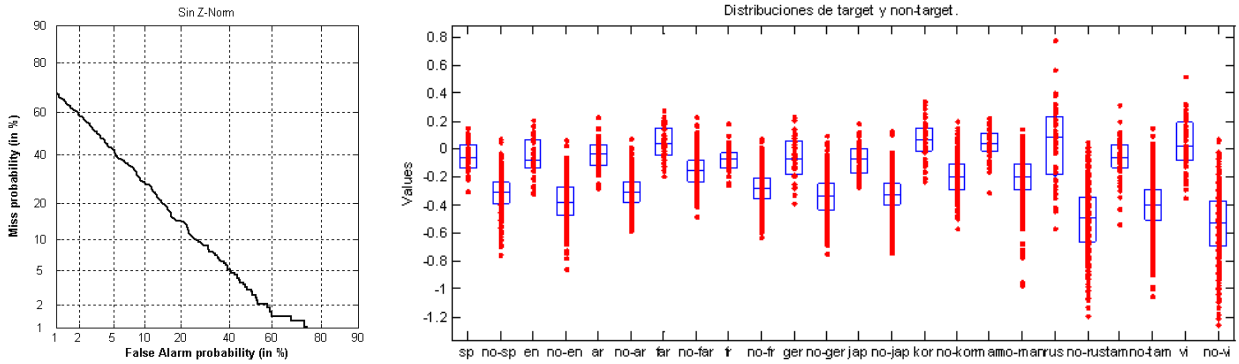

Figura 7.9: Curva DET del sistema GMM-MMI sin Z-Norm y grafica de distribución de puntuaciones.

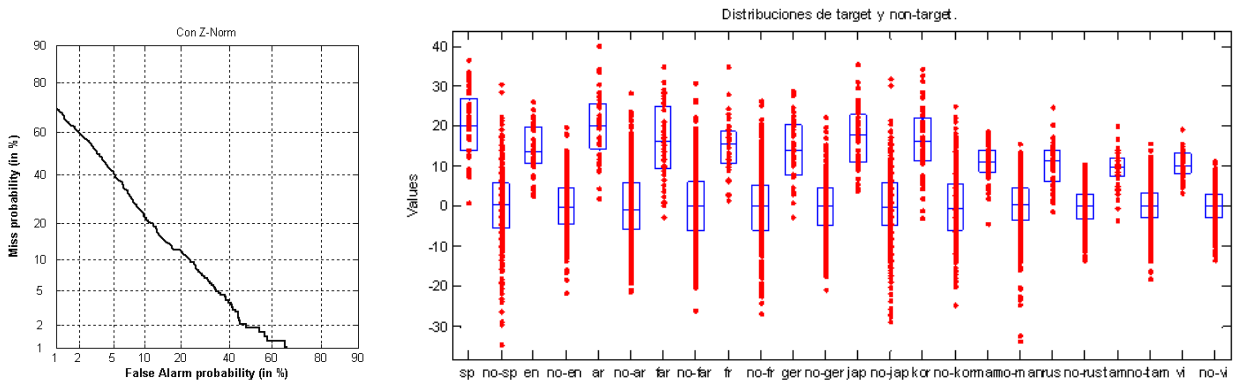

Figura 7.10: Curva DET del sistema GMM-MMI con Z-Norm y grafica de distribución de puntuaciones.

	Sistema GMM-MMI-SDC-64	Sistema GMM-MMI-SDC64 binario sin Z-Norm	Sistema GMM-MMI-SDC64 binario con Z-Norm
EER	13.95	16.87	14.92

Tabla 7.6: EER para el sistema GMM-MMI-8-SDC antes y después de aplicar Z-NORM, sobre el conjunto de evaluación de NIST 2003.

En ambos sistemas MMI y ML la aplicación de Z-Norm consigue en alinear la distribución de impostores mejorando el rendimiento global del sistema. Sin embargo, el rendimiento de este nuevo enfoque de entrenamiento es similar a entrenar un modelo por idioma. Además, los resultados de la aplicación de Z-Norm son demasiado optimistas (las distribuciones quedan alineadas perfectamente, algo difícil,sino imposible, en la práctica) por emplear los mismos ficheros de audio para evaluar los modelos que para estimar las medias y varianzas de la distribución de impostores (en la práctica se deberá reservar un conjunto del audio de entrenamiento, denominado usualmente como development, para la estimación de la distribución de las puntuaciones de los impostores de cara a la aplicación de Z-NORM).

7.1.6. Rendimiento del sistema de prueba GMM-MMI-SDC64 en NIST 2005

El sistema GMM-MMI-SDC64 fue testado con 80 ficheros por idioma obtenidos de la evaluación de NIST 2005 en la prueba de 30 segundos. En la figura 7.11 se muestra la curva DET obtenida tanto para el sistema base GMM-ML-8-SDC64 como para el sistema GMM-ML empleando los primeros 7 coeficientes cepstrales MFCC (sin el coeficiente de energia c_0). Para el entrenamiento se emplearon 40 ficheros de la evaluación de NIST 2003. Debido al escaso audio de entrenamiento, aumentar el numero de mezclas por encima de 8 degradaba el rendimiento como consecuencia de la falta de generalización. En cualquier caso el sistema GMM-MMI fue claramente superior al sistema base.

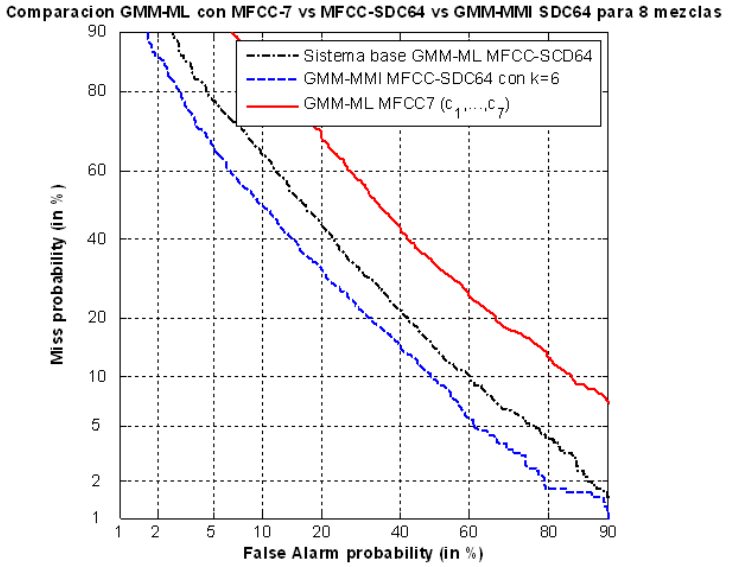

Figura 7.11: Curvas DET de los sistemas GMM-ML-8-SDC y GMM-MMI-8-SDC , sobre el conjunto de evaluación de NIST 2005.

	GMM-ML MFCC7	GMM-ML SDC64	GMM-MMI SDC64 ,k=6
EER evalset	41.37	30.00	**25.50**
EER trainset		18.58	**6.26**

Tabla 7.7: EER de los sistemas GMM-ML-8-SDC y GMM-MMI-8-SDC , sobre el conjunto de evaluación de NIST 2005.

7.2. Resultados sobre datos de evaluaciones NIST LRE'03 y LRE'05

Hasta el momento el sistema GMM-MMI ha sido probado y desarrollado en base a un conjunto reducido de audio procedente de la evaluación de NIST 2003. Mientras esto ha sido positivo en tanto que ha acortado el tiempo de implementación del mismo, también ha limitado considerablemente el alcance y generalización de los resultados obtenidos.

En esta sección se mostrará finalmente el rendimiento del sistema GMM-MMI de las figuras 5.3 (front-end) y 5.4 (back-end). Para empezar, ahora se emplearan todos los ficheros

de evaluación disponibles en NIST 2003 (en vez de 40 por idioma como se hacia antes). La tabla 7.9 muestra el número de ficheros de evaluación por idioma para NIST 2003 y NIST 2005.

	English	Spanish	Korean	Japanese	Mandarin	Hindi	Tamil	Total
NIST 03	240	80	160	80	80	80	80	80
NIST 05	990	611	365	314	972	143	183	183

Tabla 7.8: Número de ficheros de evaluación por idioma para NIST LRE 2003 y NIST LRE 2005 (prueba de 30 segundos)

El sistema GMM-MMI-128 definitivo se entrena utilizando la base de datos Callfriend. Asi mismo,el sistema de partida GMM-ML-128 se ha entrenado utilizando una hora por idioma de Callfriend balanceada por locutor (esto es, la hora de audio se obtiene de extraer la misma cantidad de cada una de las conversaciones telefónicas). El detector de silencios del sistema GMM-MMI anterior ahora se sustituye por un reconocedor fonético de húngaro. La salida del reconocedor se usa para segmentar el audio y/o extraer partes de silencio del mismo (como ya se ha mencionado con anterioridad, el criterio MMI se realiza a nivel se segmentos, no de frames como el criterio ML). Se considera segmento todo lo que hay entre dos silencios (símbolo sil en el reconocedor fonético). Los segmentos menores a 0.5 segundos no son utilizados para el entrenamiento. La parametrización empleada también es diferente a la del sistema GMM-MMI-SDC64. Ahora se emplean 7 coeficientes MFCC (sin coeficiente de energia c_0, ya que degradaba el rendimiento) concatenados con 49 coeficientes SDC 7-1-3-7 con filtrado RASTA, para formar finalmente un vector con 56 componentes.

Debido a que los idiomas de Inglés, Español y Mandarín tienen el doble de conversaciones en Callfriend, sus segmentos s serán entrenados con el factor $W_s = 0{,}5$.

La figura 7.12 muestra las curvas DET para los sistemas GMM-ML-128 y GMM-MMI-128 con una hora de entrenamiento por idioma (con scores T-normalizados) y la figura 7.13 con todo el audio de Callfriend.

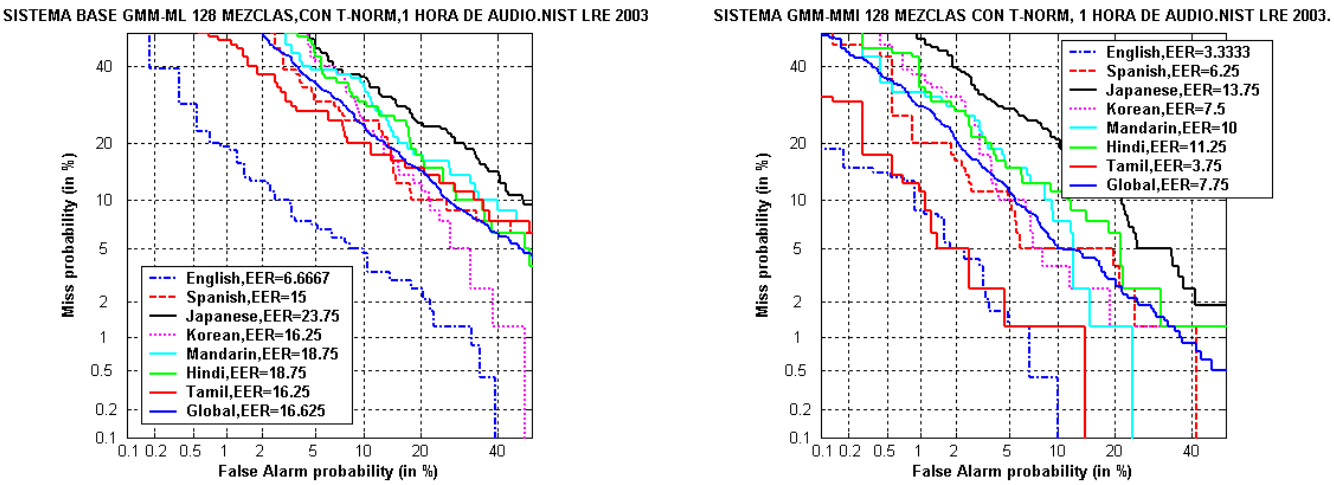

Figura 7.12: Curvas DET por idioma para los sistemas GMM-ML-128 y GMM-MMI-128 para NIST 2003. Una hora de entrenamiento.

Los resultados del sistema MMI en NIST 2003 (sobre los idiomas de 2005) se resumen en las tablas 7.9 y 7.10.

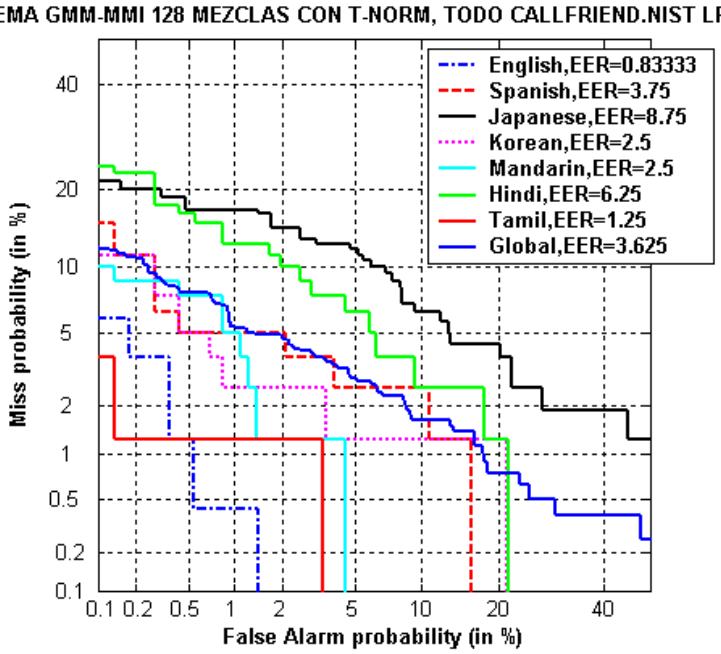

Figura 7.13: Curva DET por idioma del sistema GMM-MMI-128 para NIST 2003. Todo Callfriend.

	SISTEMA BASE GMM-ML 128, 1 HORA DE AUDIO	SISTEMA GMM-MMI 128, TODO CALLFRIEND
EER English	6.67	3.33
EER Spanish	15.00	6.25
EER Korean	23.75	13.75
EER Japanese	16.25	7.50
EER Mandarin	18.75	10.00
EER Hindi	18.75	11.25
EER Tamil	16.25	3.75
EER Global	16.62	7.75

Tabla 7.9: EER de los sistema GMM-ML-128 y GMM-ML-128 en NIST LRE 2003 (1 hora de entrenamiento).

	SISTEMA BASE GMM-ML 128, 1 HORA DE AUDIO	SISTEMA GMM-MMI 128, TODO CALLFRIEND
EER English	6.67	0.83
EER Spanish	15.00	3.75
EER Korean	23.75	8.75
EER Japanese	16.25	2.50
EER Mandarin	18.75	2.50
EER Hindi	18.75	6.25
EER Tamil	16.25	1.25
EER Global	16.62	3.62

Tabla 7.10: EER del sistema GMM-MMI-128 en NIST LRE 2003 (todo Callfriend).

Como se puede observar el EER se reduce a la mitad en el sistema MMI usando solo 1 hora de entrenamiento, y a la cuarta parte cuando se emplea todo Callfriend, frente al sistema base GMM-ML-128. Tanto para NIST 2003 y NIST 2005 se emplearon los mismos modelos de base GMM-ML-128 (por esta razón NIST 2003 solo se evaluó parcialmente en los idiomas de NIST 2005).

Los resultados sobre datos NIST 2005 se muestran en las figuras 7.14 y 7.15.

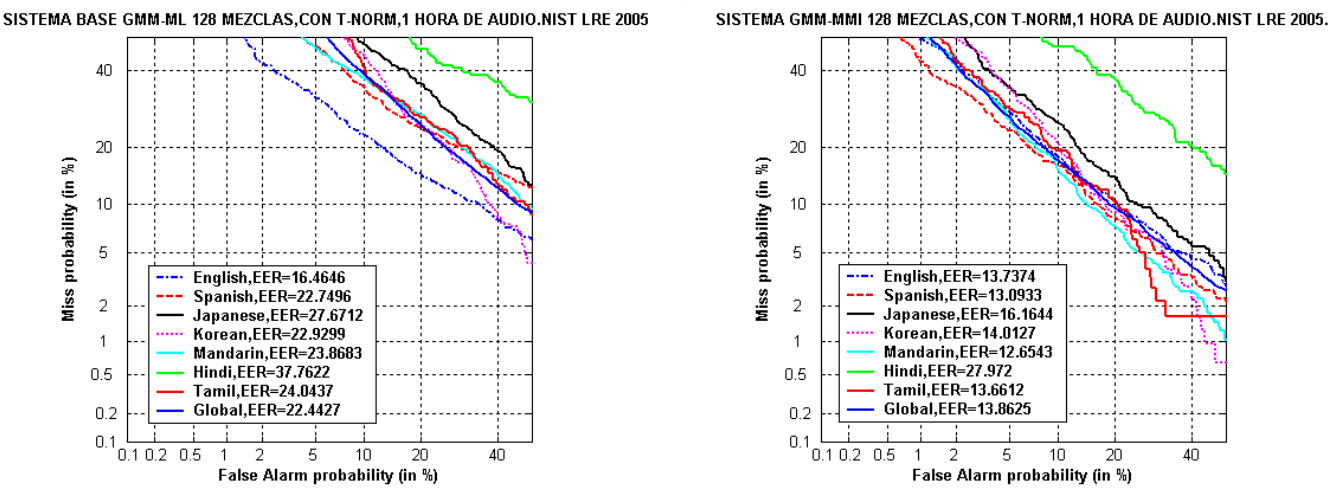

Figura 7.14: Curvas DET por idioma para los sistemas GMM-ML-128 y GMM-MMI-128 para NIST 2005. Una hora de entrenamiento.

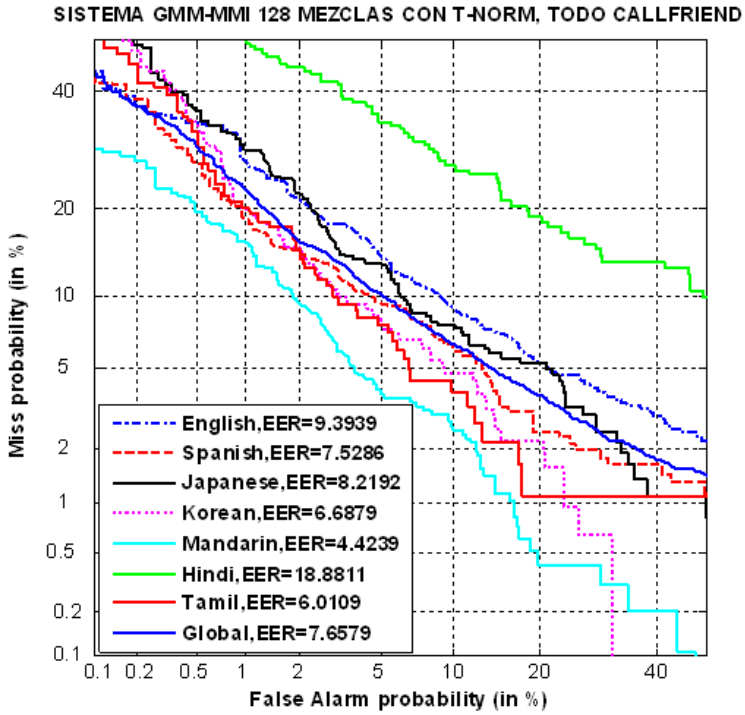

Figura 7.15: Curva DET por idioma del sistema GMM-MMI-128 para NIST 2005. Todo Callfriend.

	SISTEMA BASE GMM-ML 128, 1 HORA DE AUDIO	SISTEMA GMM-MMI 128, TODO CALLFRIEND
EER English	16.46	13.73
EER Spanish	27.74	13.09
EER Korean	27.67	16.16
EER Japanese	22.92	14.01
EER Mandarin	23.86	12.65
EER Hindi	37.76	27.97
EER Tamil	24.04	13.66
EER Global	22.47	13.86

Tabla 7.11: EER de los sistema GMM-ML-128 y GMM-ML-128 en NIST LRE 2005 (1 hora de entrenamiento).

	SISTEMA BASE GMM-ML 128, 1 HORA DE AUDIO	SISTEMA GMM-MMI 128, 1 HORA DE AUDIO
EER English	16.46	9.39
EER Spanish	27.74	7.53
EER Korean	27.67	6.69
EER Japanese	22.92	8.21
EER Mandarin	23.86	4.42
EER Hindi	37.76	18.88
EER Tamil	24.04	6.01
EER Global	22.47	7.66

Tabla 7.12: EER del sistema GMM-MMI-128 en NIST LRE 2005 (todo Callfriend).

Debido a que NIST 2005 ya no utiliza Callfriend para el audio de evaluación (en su lugar utiliza OSHU) los rendimientos son inferiores a NIST 2003 (el audio para esta última evaluación procede de una porción de Callfriend no liberada al publico).

7.3. Resultados evaluación Albayzin LV'08

Los sistemas GMM-MMI-128 y AGMM-SVM-512 participaron en la evaluación nacional de verificación de la lengua ALBAYZIN 2008. Aunque esta evaluación solo contiene 4 idiomas, la alta variabilidad de la base de datos empleada (procedente de grabaciones de programas de televisión en su mayoría) hace que tenga un nivel de dificultad similar al de las últimas evaluaciones de NIST.

7.3.1. Sistema GMM-MMI-128

El sistema GMM-MMI-128 es el mismo que el descrito en la sección anterior. Para la evaluación de Albayzin solo se dispone de aproximadamente 6 horas (después de eliminar los silencios) para su entrenamiento.

La figura 7.16 muestra la curva DET del sistema base GMM-ML con 128 mezclas, utilizando 6 horas de audio para el entrenamiento. En la figura 7.17 se muestra el rendimiento del sistema GMM-MMI-128 después de 20 iteraciones por BWE. Para el algoritmo BWE se utilizó una constante de escalamiento de $C = 6$ (valor óptimo). Las figuras 7.18 y 7.19 muestran las mismas curvas después de hacer T-NORM.

La figura 7.20 muestra el rendimiento global del sistema GMM-MMI-128 sin T-NORM.

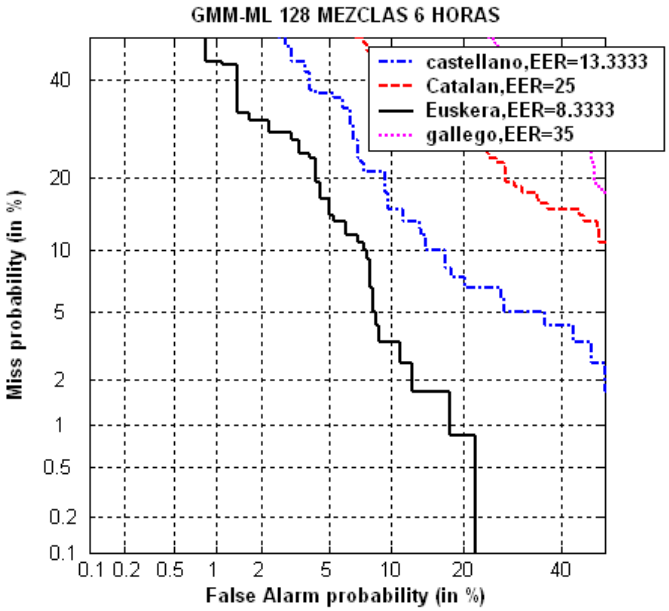

Figura 7.16: Curvas DET por idioma del sistema base GMM-ML-128 sin T-NORM.

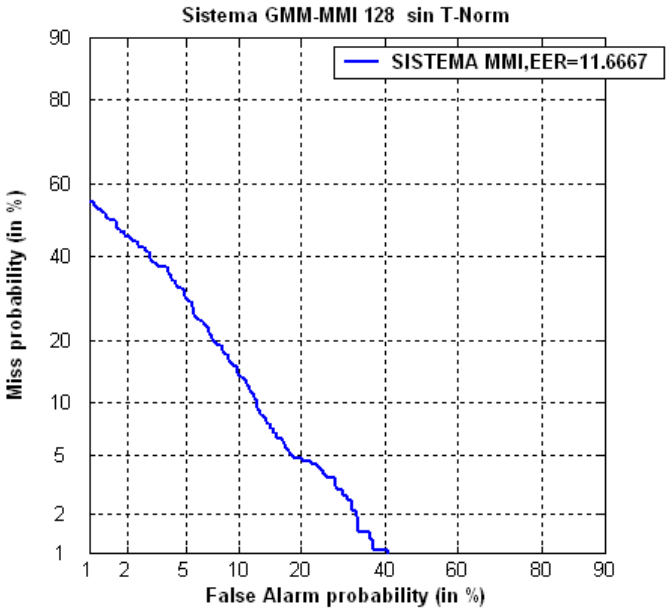

Figura 7.20: Curva DET global para el sistema GMM-MMI-128 sin T-NORM.

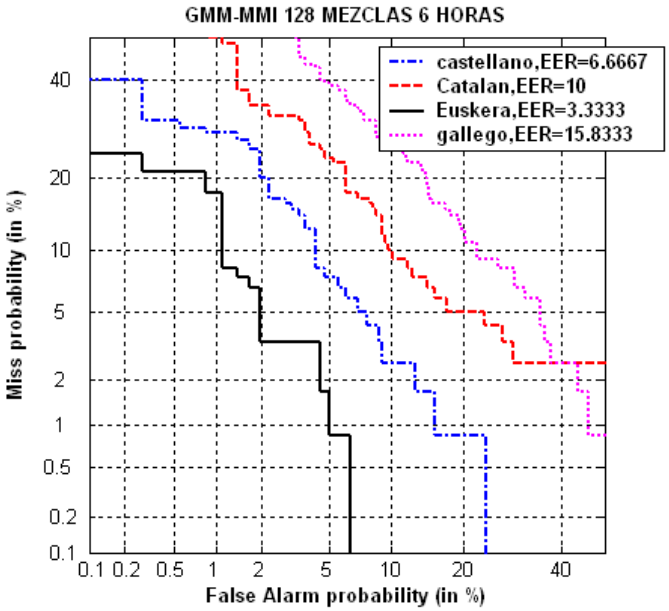

Figura 7.17: Curvas DET por idioma para el sistema GMM-MMI-128 sin T-NORM.

Resumen de resultados:

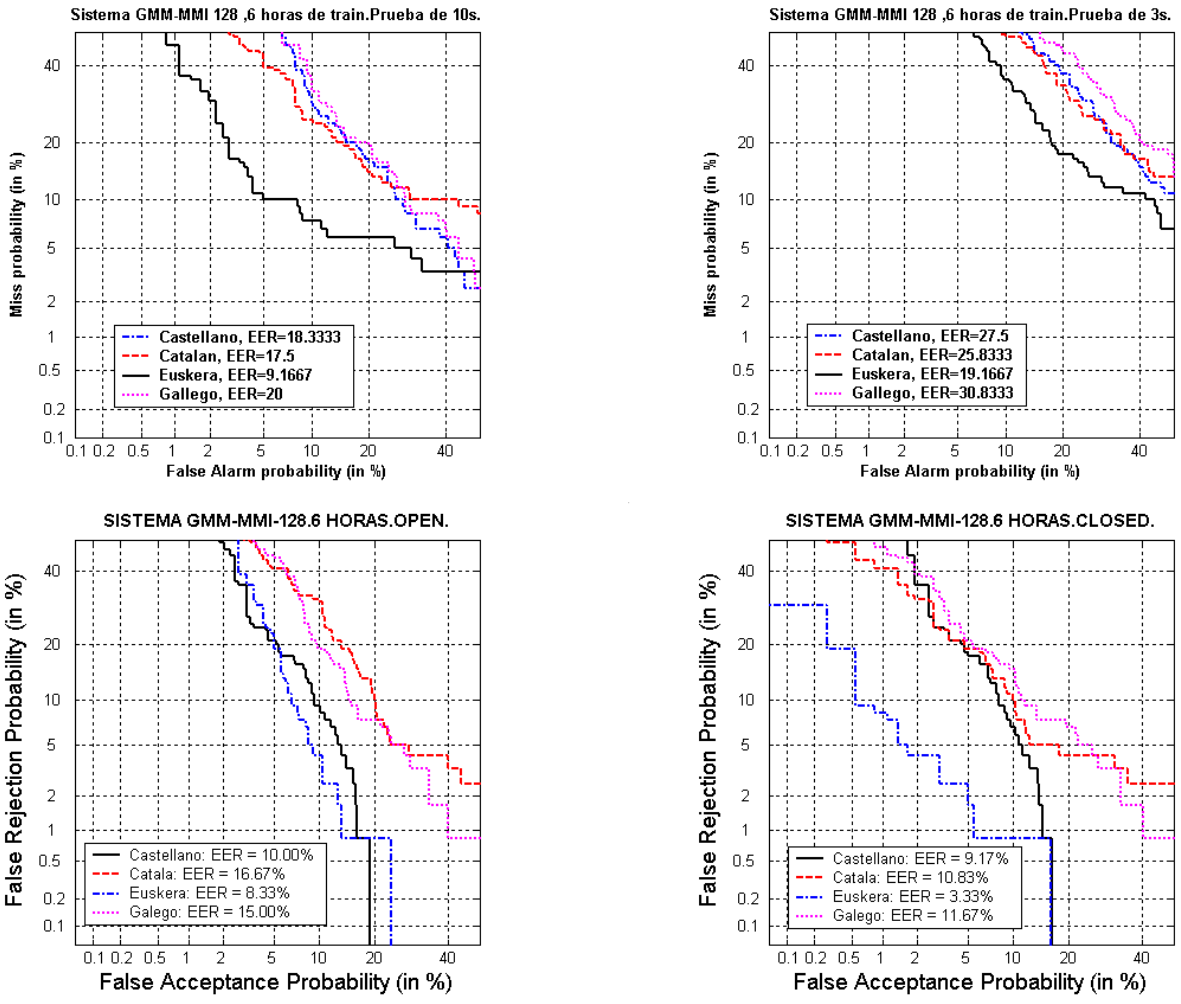

Figura 7.21: Pruebas de 30, 10 y 3 segundos en conjunto abierto y de 30 s en conjunto cerrado.

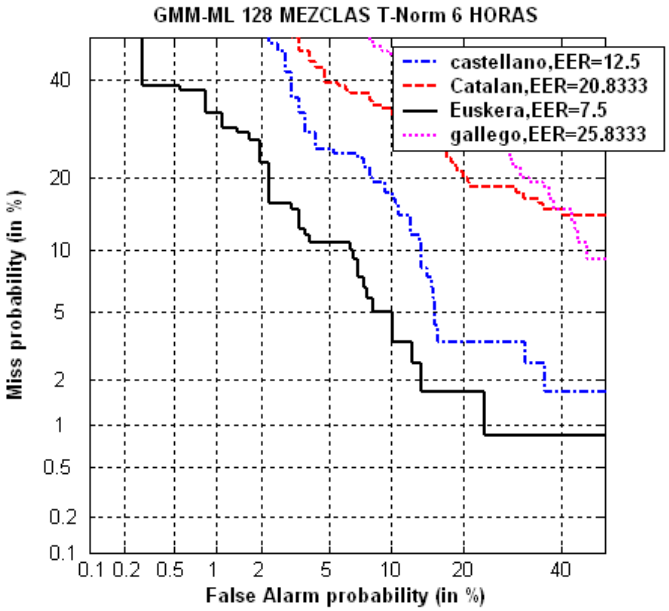

Figura 7.18: Curvas DET por idioma del sistema base GMM-ML-128 con T-NORM.

Sistema GMM-MMI 128	Prueba de 30 s	Prueba de 10 s	Prueba de 3 s
EER Castellano	9.17	18.33	27.50
EER Catalán	10.83	17.50	25.83
EER Eusquera	3.33	9.16	19.16
EER Gallego	11.67	20.00	30.83

Tabla 7.13: Valores EER del sistema GMM-MMI-128 en Albayzin LV 2009. Pruebas de 30, 10, y 3 segundos en conjunto cerrado.

Aunque no se muestra la curva DET, la segunda columna de la tabla de a continuación indica los valores de EER por idioma para el audio de evaluación que se terminó usando finalmente en la competición.

Sistema GMM-MMI-128	Prueba de 30 s CLOSED DEV	Prueba de 30 s CLOSED EVAL
EER Castellano	9.17	13.33
EER Catalán	10.83	11.66
EER Eusquera	3.33	7.50
EER Gallego	11.67	15.83

Tabla 7.14: Valores EER del sistema GMM-MMI-128 para las pruebas de 30 segundos (audio de evaluación de desarrollo y audio final de la evaluación.

7.3.2. Influencia del sistema base en GMM-MMI

Para ver la influencia en el rendimiento del sistema MMI de los GMM's iniciales, se omitió la fase de entrenamiento de los modelos GMM por ML como punto de partida para el entrenamiento MMI. En su lugar los modelos de partida para MMI procederán de dos iteraciones

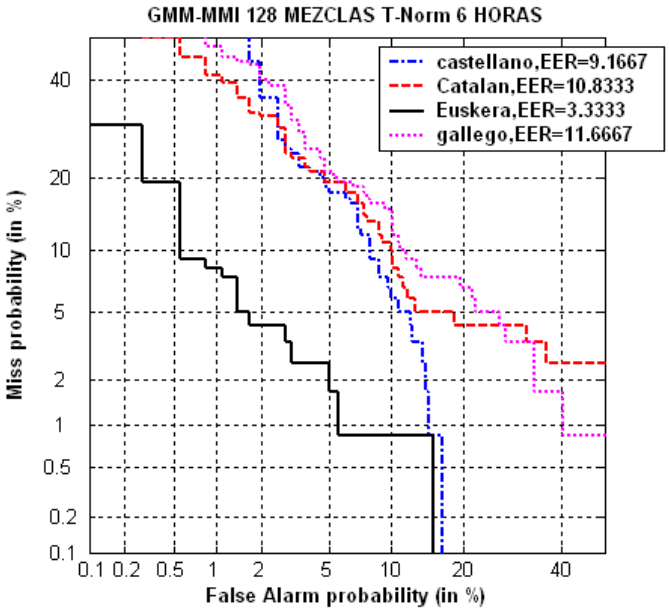

Figura 7.19: Curvas DET por idioma para el sistema GMM-MMI-128 con T-NORM.

de K-means (con 128 centroides). Las curvas DET globales para los GMM's inicializados por K-means sin y con T-Norm son las siguientes:

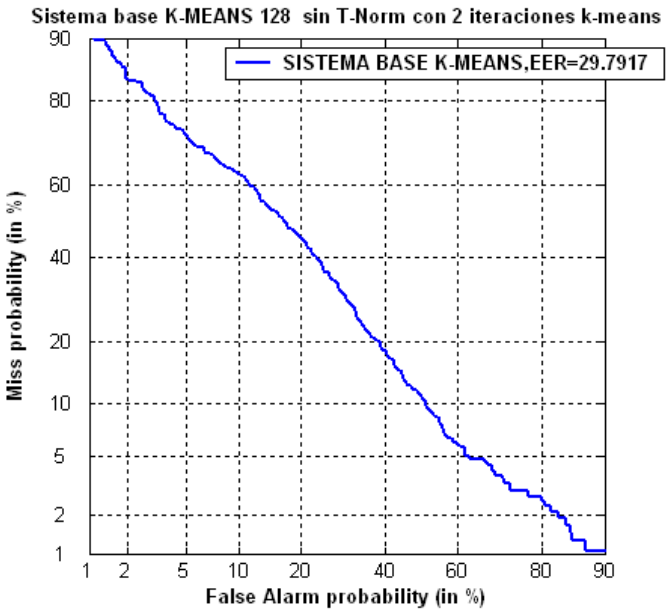

Figura 7.22: Curva DET global para el sistema base K-Means-128 (2 iteraciones) sin T-NORM.

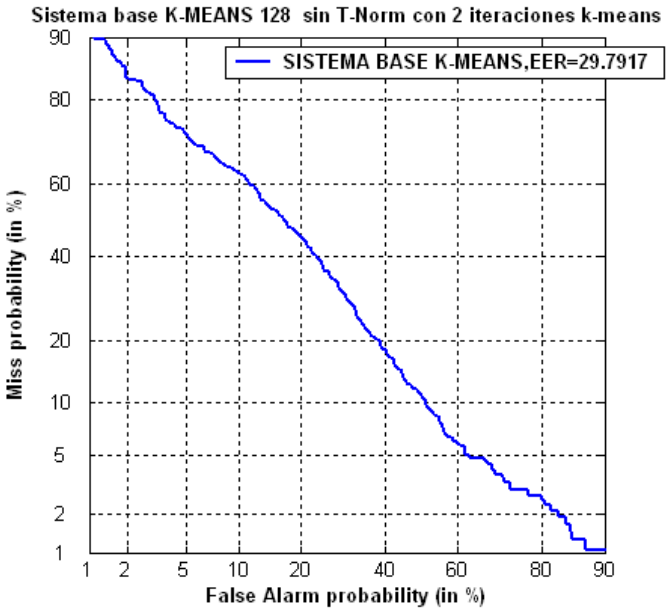

Figura 7.23: Curva DET global para el sistema base K-Means-128 (2 iteraciones) sin T-NORM.

Y la curva DET del sistema GMM-MMI partiendo de dichos modelos es:

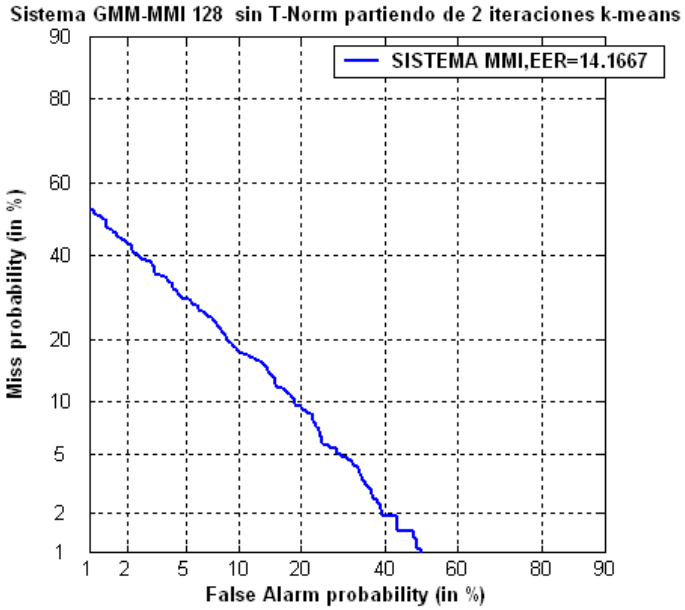

Figura 7.24: Curva DET global para el sistema GMM-MMI-128 partiendo de K-Means-128 (2 iteraciones) sin T-NORM.

Como se puede observar el EER se degrada en 3 puntos frente al sistema inicializado con ML, pero no tanto como cabria esperar.

7.3.3. Sistema AGMM-SVM-512

El sistema AGMM-SVM se creó con la idea de poder comparar los resultados del sistema discriminativo GMM-MMI con otro sistema que también fuera discriminativo.Inesperadamente, el sistema AGMM-SVM finalmente llegó incluso a superar en rendimiento al sistema MMI, y fue, de hecho, el sistema con mejor rendimiento que concurrió a la evaluación.

El sistema AGMM-SVM esta descrito detalladamente en la sección 5.2.1.2.

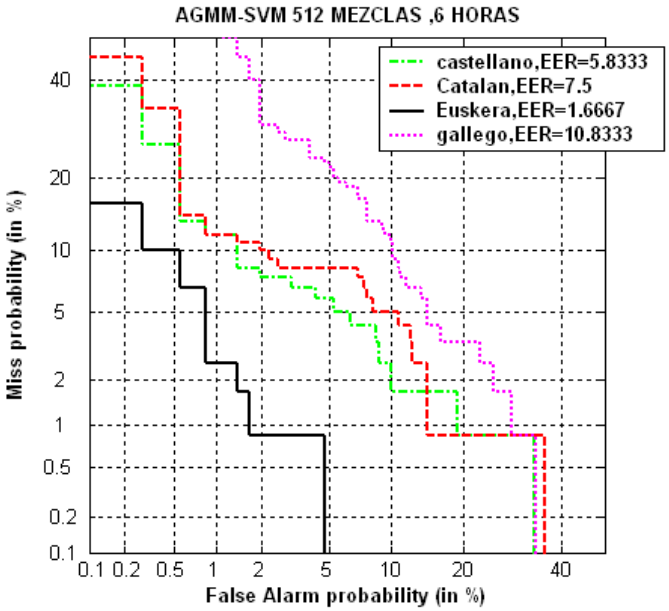

Figura 7.25: Curvas DET por idioma del sistema AGMM-SVM-512 sin T-NORM.

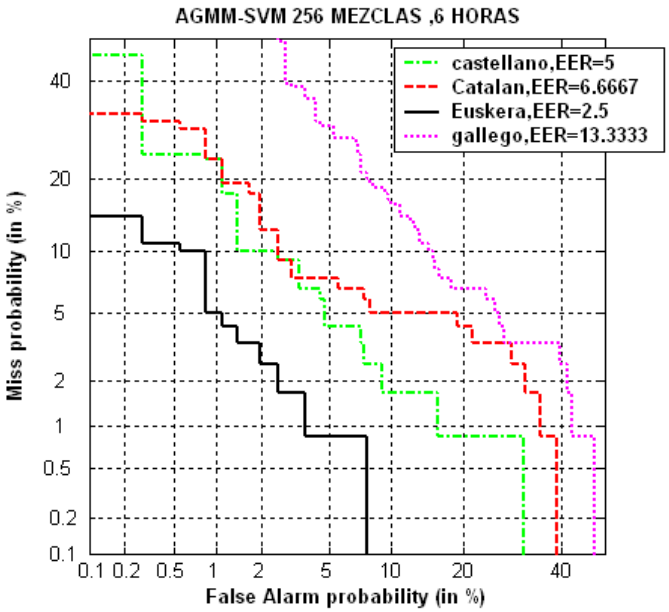

Figura 7.26: Curvas DET por idioma del sistema AGMM-SVM-256 sin T-NORM.

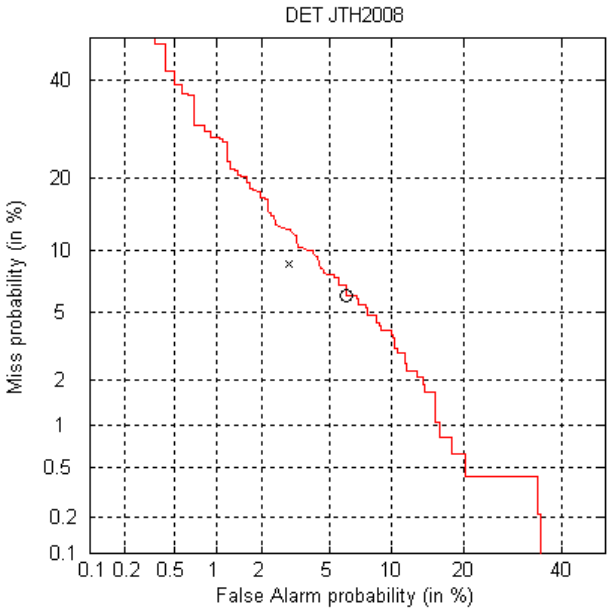

Figura 7.27: Curvas DET global del sistema AGMM-SVM-512 sin T-NORM obtenida con el software oficial de Albayzin LV 2008.

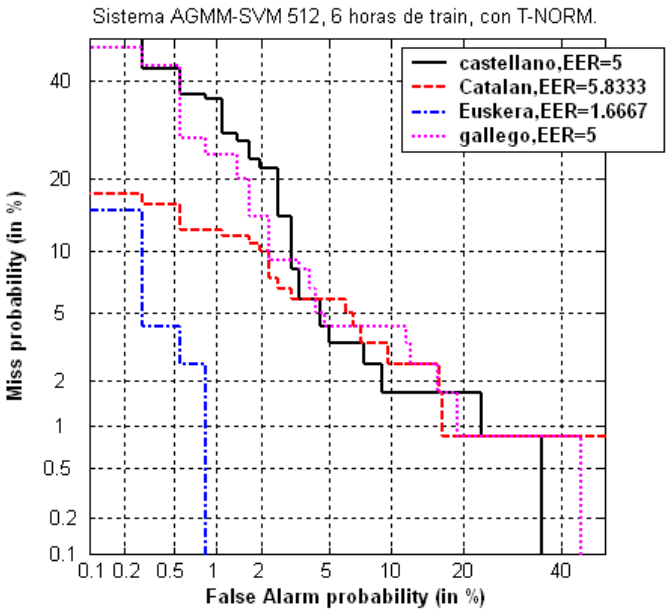

Figura 7.28: Curvas DET por idioma del sistema AGMM-SVM-512 con T-NORM.

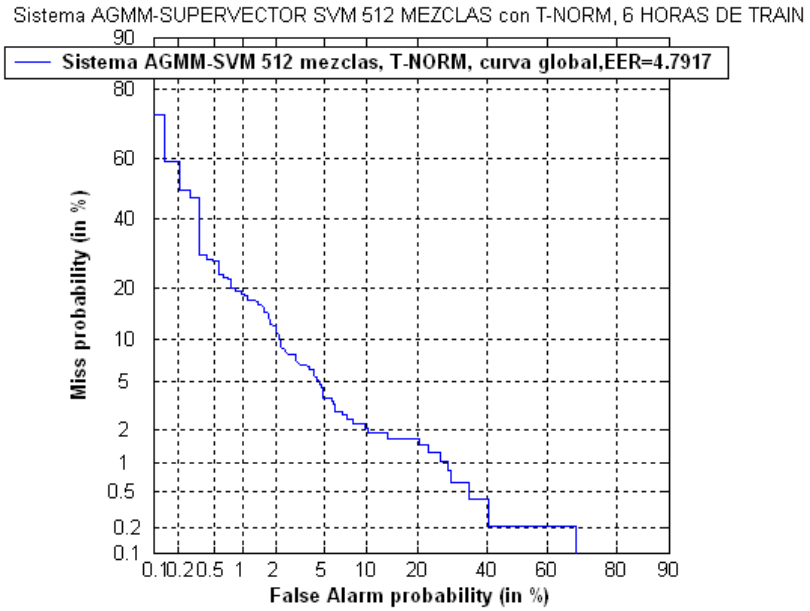

Figura 7.29: Curvas DET global del sistema AGMM-SVM-512 con T-NORM.

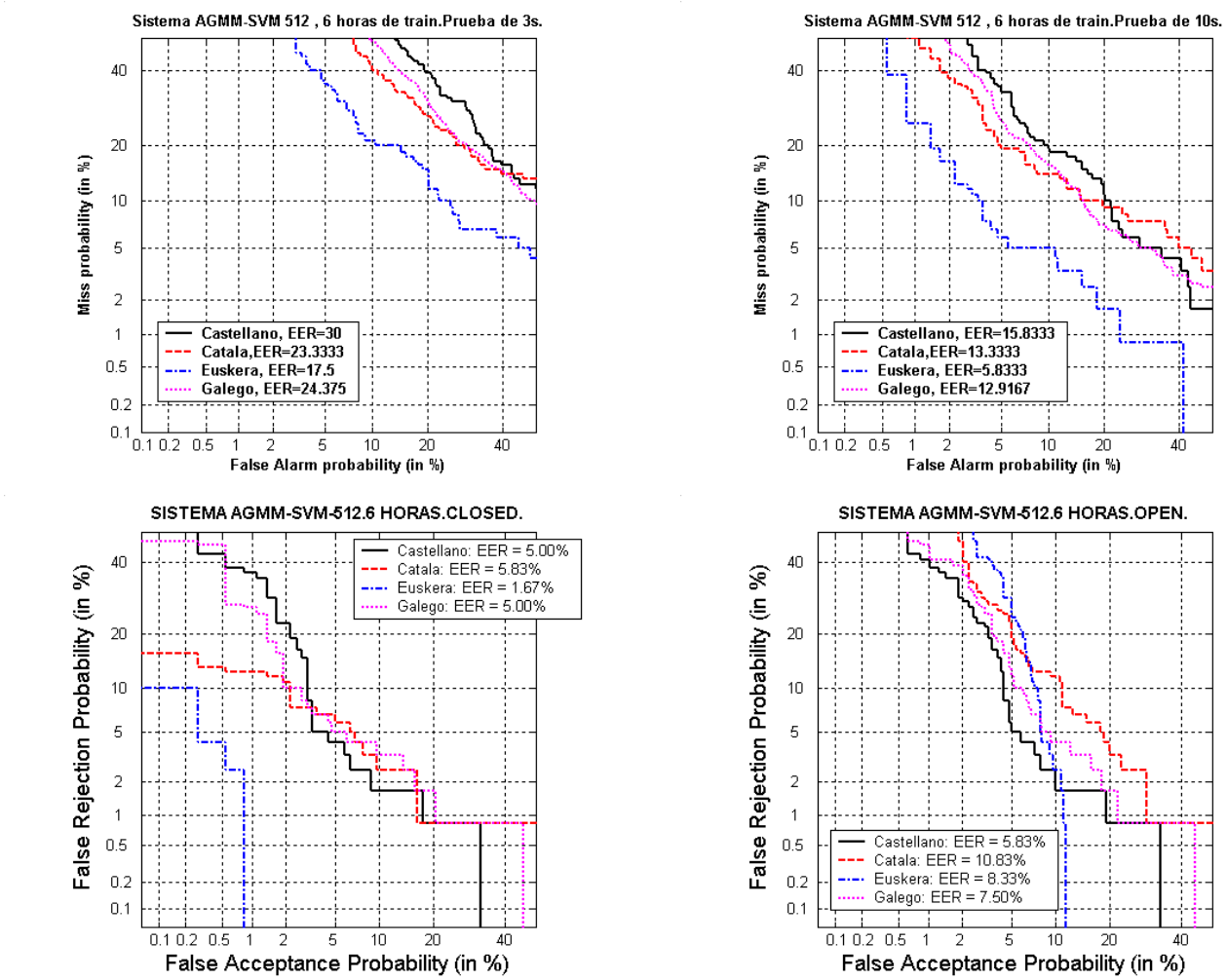

Figura 7.30: Curvas DET por idioma para el sistema AGMM-SVM-512 en Albayzin LV 2008. Pruebas de 30, 10 y 3 segundos en conjunto abierto y de 30 s en conjunto cerrado.

Sistema AGMM-SVM 512	Prueba de 30 s	Prueba de 10 s	Prueba de 3 s
EER Castellano	5.00	15.83	30.00
EER Catalán	5.83	13.33	23.33
EER Eusquera	1.67	5.83	17.50
EER Gallego	5.00	12.91	24.37

Tabla 7.15: Valores EER del sistema AGMM-SVM-512 en Albayzin LV 2009. Pruebas de 30, 10, y 3 segundos en conjunto cerrado.

Sistema AGMM-SVM 512	Prueba de 30 s CLOSED	Prueba de 30 s OPEN
EER Castellano	5.00	5.83
EER Catalán	5.83	10.83
EER Eusquera	1.67	8.33
EER Gallego	5.00	7.50

Tabla 7.16: Valores EER del sistema AGMM-SVM-512 para las pruebas de 30 segundos en conjunto cerrado y abierto.

Sistema AGMM-SVM 512	Prueba de 30 s CLOSED DEV	Prueba de 30 s CLOSED EVAL
EER Castellano	5.00	8.33
EER Catalán	5.83	3.33
EER Eusquera	1.67	4.16
EER Gallego	5.00	10.83

Tabla 7.17: Valores EER del sistema AGMM-SVM-512 para las pruebas de 30 segundos (audio de evaluación de desarrollo y audio final de la evaluación.

Las clavés para que el sistema AGMM-SVM tenga tan buen rendimiento se enumeran a continuación:

- Tipo de Kernel : el kernel que dio el mejor rendimiento fue el siguiente:

$$\frac{\mu_{id} - v_{id}}{\sigma_{id}} + v_{id} \tag{7.8}$$

- Segmentacion de los ficheros de entrenamiento. Los ficheros de entrenamiento se dividieron en segmentos de 30 segundos, de esta forma el rendimiento mejora increiblemente, principalmente por dos motivos :

  1. Al dividir los ficheros tenemos mas supervectores con los que entrenar .Es algo obvio que un sistema funcionara mejor cuando mas items le proporcionamos(aprendizaje supervisado). La limitación está en la cantidad de memoria RAM disponible. Por ejemplo para la evaluación de Albayzin donde se dispone de 8 horas por idioma se requerirán en total $8*3600/30 = 28{,}800$ supervector. Como cada supervector tiene 512 componentes almacenadas como double (64 bits =8 bytes) durante el entrenamiento se requeriría almacenar en memoria RAM un total de $(4) * 28{,}800 * 512 * 8$ bytes, aproximadamente 0.5 GBytes . Una evaluación de idioma como NIST que tiene 12 idiomas y disponemos de 40 horas (Callfriend) para entrenar cada modelo necesitará una maquina con procesador de 64 bits de al menos 8 GBytes de RAM. Por suerte un modulo DDR2 de 2 gigas cuesta 30 euros (en el año 2008).
  2. El tamaño de los ficheros es adaptado al tamaño de los ficheros de evaluación (para la prueba de 30 segundos). Esto último no esta del todo claro como influye.

7.3.4. Fusión suma AGMM-SVM-512 y AGMM-SVM-256

Aunque lo normal es fusionar sistemas poco correlacionados [14][15] (normalmente basados en niveles de información diferentes,como ejemplo un PRLM con un GMM), se intentó fusionar el sistema GMM-MMI con el sistema AGMM-SVM sin éxito. Entonces se probó a fusionar los

AGMM-SVM de 512 y 256 mezclas, obteniendo un ligero aumento de rendimiento. La figura, a continuación , muestra la curva DET del sistema resultante de sumar las puntuaciones de ambos sistemas con resultados positivos.

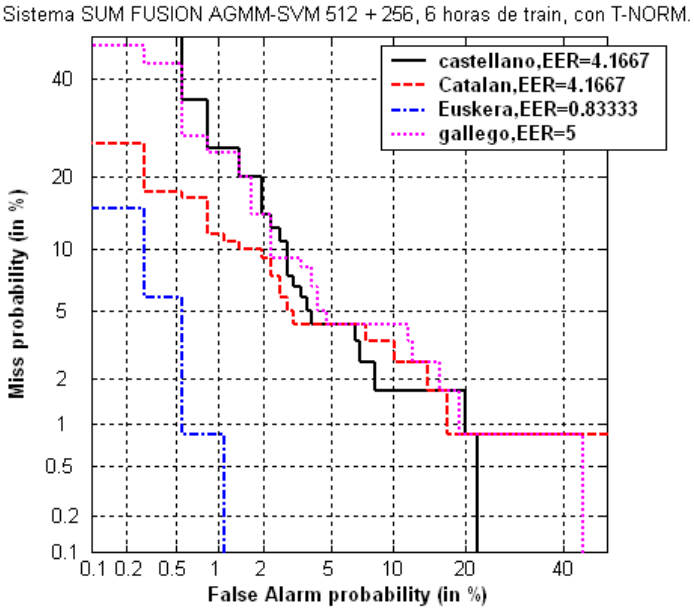

Figura 7.31: Curvas DET por idioma de la fusión de los sistemas AGMM-SVM-512 y AGMM-SVM-256 con T-NORM.

7.3.5. Resultados Albayzin LV 2008 conjunto de evaluación

Las siguientes figuras muestran las curvas DET para la prueba de 30 segundos en conjunto cerrado utilizando el audio final de la competición. Se puede observar como el rendimiento de los sistemas se degrada en torno a un 25 %, respecto al audio de evaluación de desarrollo.

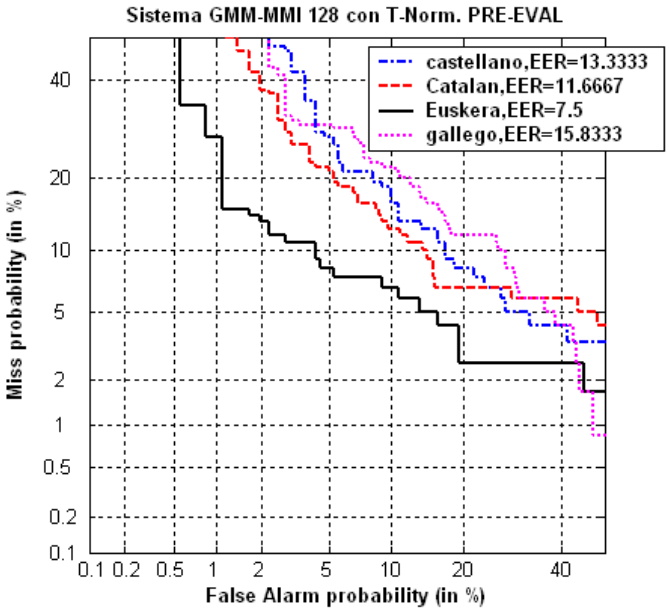

Figura 7.32: Curvas DET por idioma del sistema GMM-MMI-128 en el conjunto de evaluación definitivo.

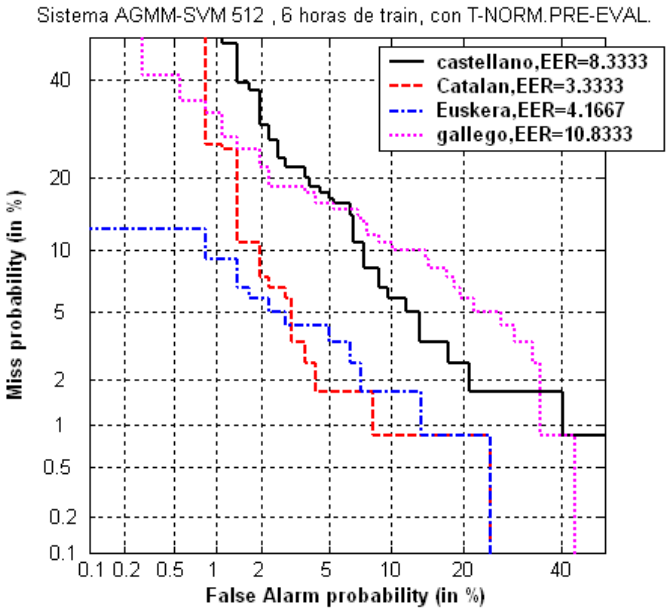

Figura 7.33: Curvas DET por idioma del sistema AGMM-SVM-512 en el conjunto de evaluación definitivo.

7.3.6. Comparativa GMM-ML vs GMM-MMI vs AGMM-SVM

Se comparan las curvas DET globales de los tres sistemas primarios. Como se puede ver el sistema MMI reduce en un 50 % el EER del sistema base ML, y a su vez el sistema AGMM-SVM reduce en otro 50 % el EER del MMI.

	GMM-ML 128	GMM-MMI 128	AGMM-SVM-512
EER GLOBAL	17.08	9.16	4.79

Tabla 7.18: EER de los sistemas GMM-ML-128, GMM-MMI-128 y AGMM-SVM-512 para la prueba de 30 segundos.

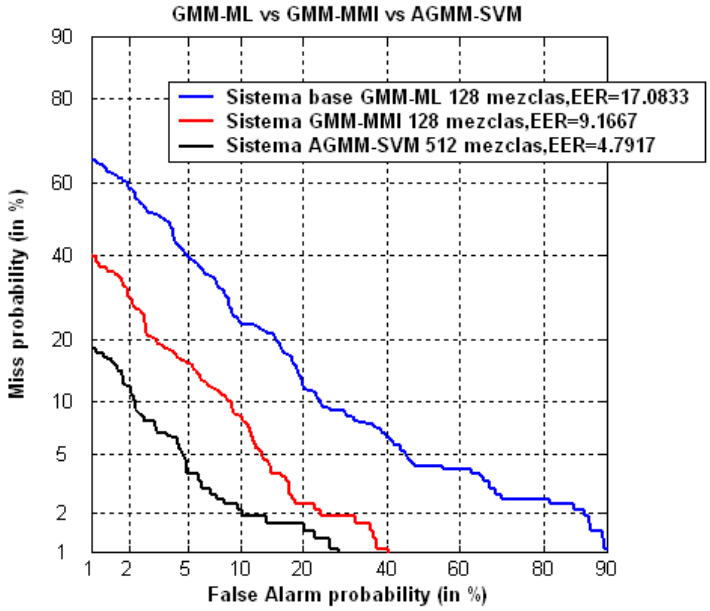

Figura 7.34: Curvas DET globales para los sistemas GMM-ML-128, GMM-MMI-128 y AGMM-SVM-512.

7.3.7. Tiempos de ejecución de los sistemas GMM-MMI y AGMM-SVM

En la tabla siguiente se resumen los tiempos de ejecución de todos los sistemas presentados por el ATVS, y en particular de los sistemas GMM-MMI y AGMM-SVM. Las especificaciones de la CPU son : Dual-Core AMD Opteron(TM) modelo 1214 , a 2.2 GHz. Los tiempos del sistema GMM-MMI, por ejemplo, se calcularon de la siguiente manera:

- Parametrización : Igual que la entrada de la tabla del sistema SVM-GMM SDC multiplicada por un factor de 56/49.
- Training (aprox): Tiempo training GMM-128 + 20 iteraciones MMI = 1 hora + 4*20*40(min/(iteración*procesador))=54.3333 horas. El factor de 4 se debe a que se utilizó 4 procesadores en paralelo pero hay que darlo con un solo núcleo.
- Scoring para prueba 30 s (600 ficheros) (aprox): 500 segundos/3600s=0.1388 horas. Como el tiempo de scoring es proporcional a la duración , el tiempo total será de 0.1388 + 0.1388*10/30 + 0.1388*3/30=0.1990 horas.

ATVS System Times (hours)							
	SVM-GLDS		GMM-SVM		AGMM-SVM	GMM-MMI	Phone-SVM
	MFCC	SDC	MFCC	SDC			
16KHz a 8KHz	0.343	(0,343)	(0,343)	(0,343)	(0.343)	(0.343)	(0,343)
MFCC	0.187		(0.187)				0,444
SDC		0.185		(0.185)	(0.211)	(0.211)	
Phone Rec.							6,997
Training							
Expansion	2.288	1.096					
Scoring	0.378	0.434	0.556	0.505	0.577	0.199	0.972
Total by Subsystem	3.196	2.058	1.086	1.033	1.131	55.086	8.756
Speed by Subsystem	0.438	0.282	0.149	0.141	0.1611	7.540	1.200

Tabla 7.19: Tiempos de evaluación de los sistemas presentados a la evaluación ALBAYZIN

7.4. Resultados de la técnica Eigenchannel Compensation

Eigenchannel Compensation [37][39][40] intenta reducir cualquier fuente de variabilidad dentro de un mismo idioma (distintos locutores, distinto contenido, distintos canales y condiciones) utilizando Factors Analysis. Esta variabilidad es la responsable de degradar el rendimiento durante la evaluación. La versión que se ha implementado, se aplica a nivel de vectores de características, esto es, a nivel paramétrico (ya ha sido descrita en el capitulo del estado del arte).

La figura 7.35 se obtuvo de la siguiente forma:

1. Se entrena un modelo UBM de 32 mezclas utilizando todo el audio disponible.

2. Para cada fichero de entrenamiento se obtiene un supervector concatenando las medias del GMM obtenido de la adaptación por MAP de dicho fichero con un factor de adaptación de $r = 14$. Las medias de dividen (normalizan) por la raiz cuadrada de la varianza del UBM (esto es, la desviación típica). Los supervectores son almacenados en un fichero.

3. Se aplica Eigenchannel compensation en el dominio de la features utilizando el UBM, el fichero anterior y la la lista de ficheros de train y evaluación a compensar. La ventaja de realizar la compensación en el dominio cepstral es la posibilidad de utilizar cualquier tipo de clasificador (GMM,MMI,GMM-SVM). Se utilizan $R = 40$ autocanales.

4. Utilizando los vectores de características compensados se entrena un UBM-GMM de 64 mezclas para cada idioma (debido a que se dispone de tan poco audio para entrenar,no es recomendable un número de mezclas mayor). Aquí se podía haber usado otro tipo de clasificador. También se compara el mismo sistema pero usando los parámetros originales sin compensar (parametrización 7 MFCC + 7-1-3-7 SDC con RASTA, igual que la del sistema MMI).

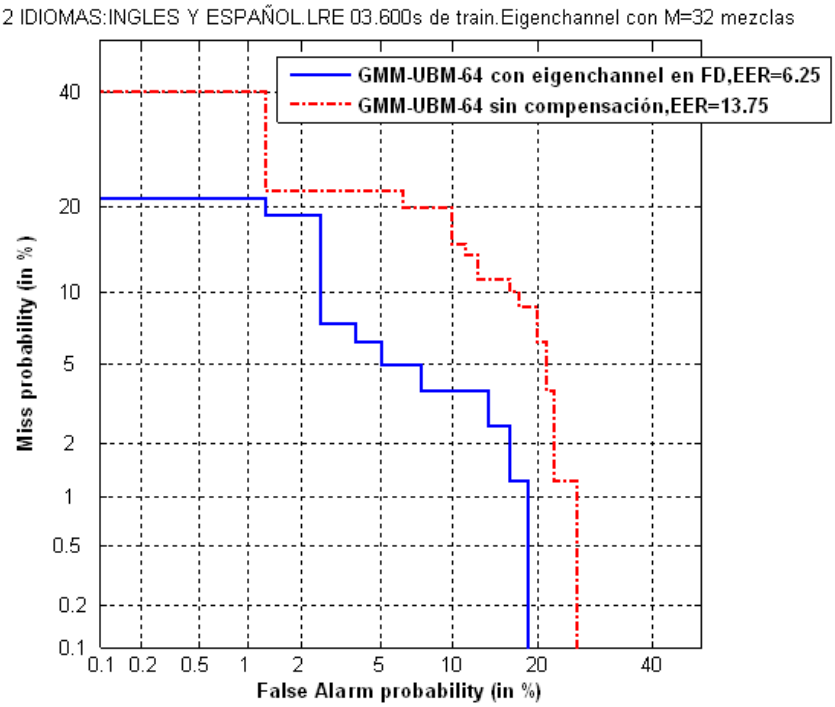

Figura 7.35: Curvas DET globales de español e inglés antes y después de aplicar eigenchannel en NIST 2003.

Igualmente, las figuras 7.36, 7.37, y 7.38 se obtuvieron entrenando la matriz de autocanal con 80 ficheros de Callfriend por idioma y R=50 autocanales.

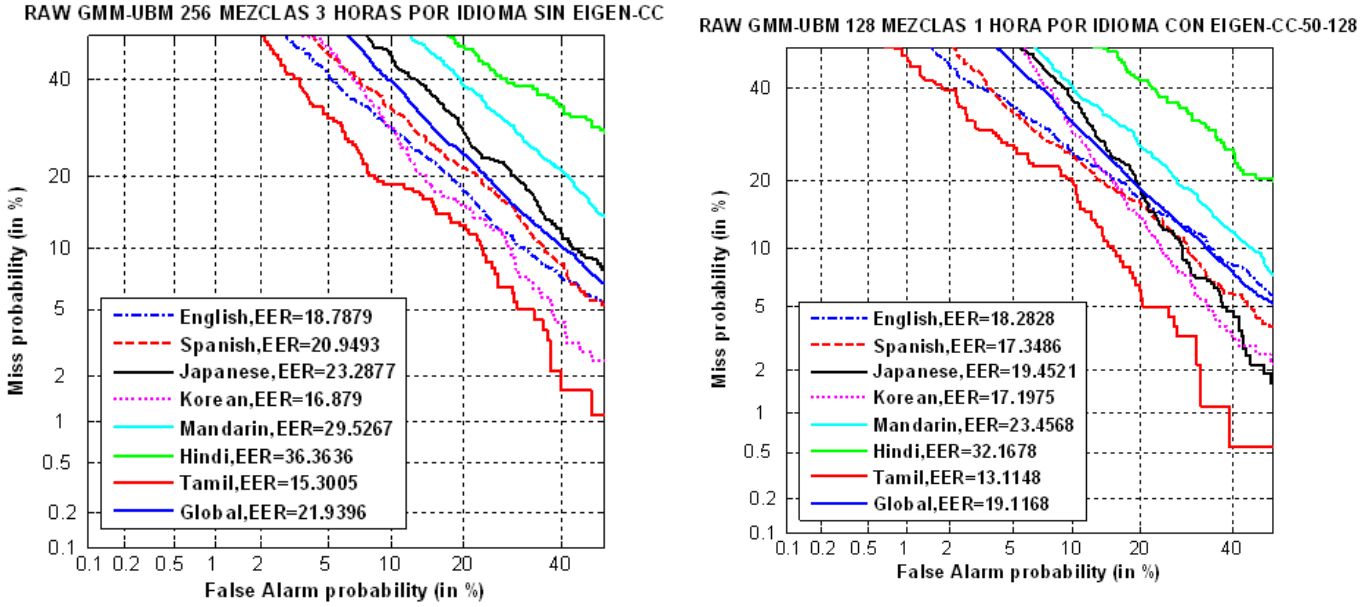

Figura 7.36: Curvas DET por idioma para los sistemas GMM-UBM-256 sin eigenchannel y GMM-UBM-128 con eigenchannel en NIST LRE 2005.

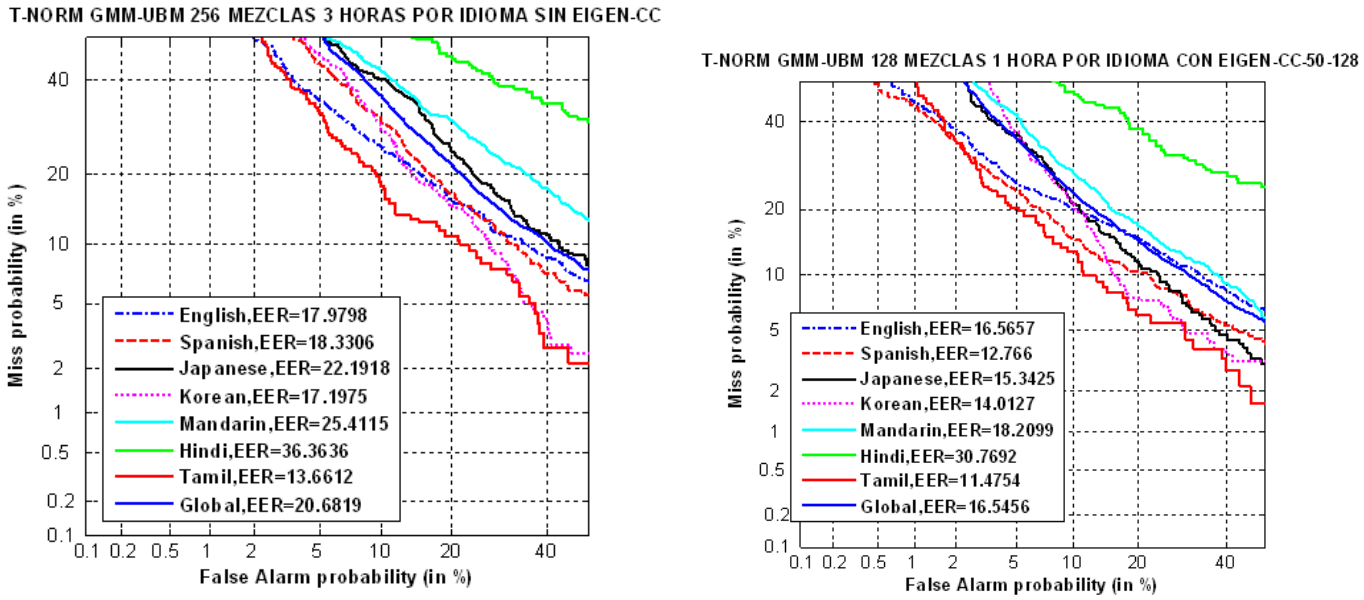

Figura 7.37: Curvas DET por idioma para los sistemas GMM-UBM-256 sin eigenchannel y GMM-UBM-128 con eigenchannel en NIST LRE 2005. Scores con T-NORM.

Los resultados pueden mejorarse mucho más empleando mas ficheros y mezclas para obtener la matriz de autocanal. En las tablas siguientes se resumen los resultados anteriores:

	RAW GMM-UBM 256 MEZCLAS, 3 HORAS SIN EIGEN-CC	RAW GMM-UBM 128 MEZCLAS, 1 HORA POR IDIOMA CON EIGEN-CC-128
EER English	18.78	18.28
EER Spanish	20.94	17.34
EER Korean	23.28	19.45
EER Japanese	16.87	17.19
EER Mandarin	29.52	23.45
EER Hindi	36.36	32.16
EER Tamil	15.30	13.11
EER Global	21.93	19.11

Tabla 7.20: Valores de EER de los sistemas GMM-UBM-256 sin eigenchanel y GMM-UBM-128 con eigenchannel compensation en NIST LRE 2005.

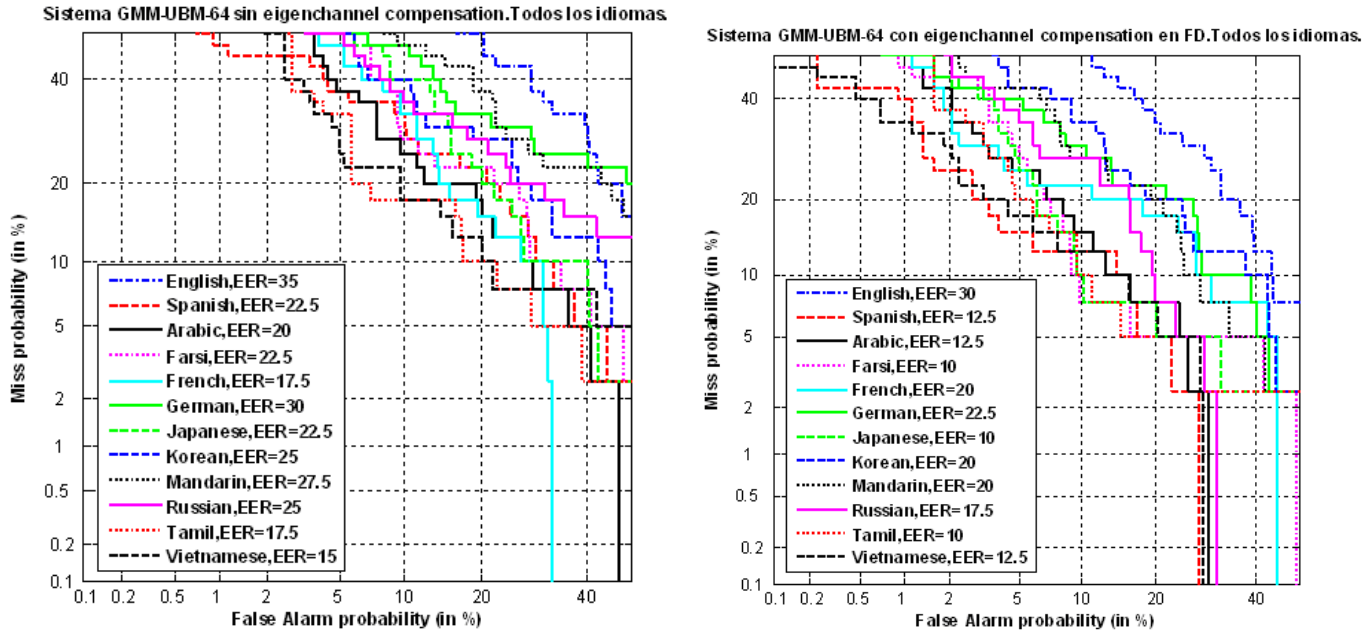

Figura 7.38: Curvas DET por idioma para los sistemas GMM-UBM-64 sin eigenchannel y GMM-UBM-64 con eigenchannel en NIST LRE 2003.

	T-NORM GMM-UBM 256 MEZCLAS, 3 HORAS SIN EIGEN-CC	T-NORM GMM-UBM 128 MEZCLAS, 1 HORA POR IDIOMA CON EIGEN-CC-128
EER English	17.97	16.56
EER Spanish	18.33	12.76
EER Korean	22.19	15.34
EER Japanese	17.19	14.01
EER Mandarin	25.41	18.20
EER Hindi	36.36	30.76
EER Tamil	13.66	11.47
EER Global	20.68	16.54

Tabla 7.21: Valores de EER de los sistemas GMM-UBM-64 sin eigenchanel y GMM-UBM-64 con eigenchannel compensation en NIST LRE 2003.

	GMM-UBM-64 SIN EIGEN-CC	GMM-UBM-64 CON EIGEN-CC
EER English	35.00	30.00
EER Spanish	22.50	12.50
EER Arabic	20.00	12.50
EER Farsi	22.50	10.00
EER French	17.50	20.00
EER German	30.00	22.50
EER Japanese	22.50	10.00
EER Korean	25.00	20.00
EER Mandarin	27.50	20.00
EER Russian	25.00	17.50
EER Tamil	17.50	10.00
EER Vietnamese	15.00	12.50

Tabla 7.22: Valores de EER de los sistemas GMM-UBM-256 sin eigenchanel y GMM-UBM-128 con eigenchannel compensation en NIST LRE 2005. Scores T-Normalizados.

8

Conclusiones

Tradicionalmente se ha utilizado el criterio generativo de máxima verosimilitud (ML) para obtener los modelos de idioma. El criterio ML garantiza que los reconocedores de idioma entrenados con él serán óptimos siempre que se disponga de una cantidad de audio ilimitada para entrenar los modelos, y, más importante si cabe, que se conozca la forma funcional de la distribución de probabilidad de los datos de entrenamiento (la cual, normalmente se asume Gaussiana).

Lamentablemente, ninguna de las dos hipótesis anteriores se mantiene en la práctica. Esto ha llevado a los investigadores a desarrollar nuevos y más flexibles criterios de entrenamiento (y presumiblemente de mayor precisión en el reconocimiento, que es el objetivo último) para entrenar los reconocedores de idioma. Puesto que la tarea básica del reconocimiento de idioma es determinar y/o verificar la identidad del idioma contenido en una porción de audio (estimando $P(L|O)$), un buen reconocedor debería elegir el idioma L que reduzca la cantidad de incertidumbre de la respuesta correcta. Matemáticamente esto equivale a minimizar la entropía condicional $H(L|O)$, que a su vez, como se vio en la sección 4.1, equivale a maximizar la información mutua entre el idioma correcto y el audio que lo contiene, esto es, $I(O;L)$.

La mayor flexibilidad del criterio MMI (al no requerir conocer la función de distribución de los datos) trae consigo una mejora en la precisión, pero a su vez complica el entrenamiento, siendo este último uno de los mayores inconvenientes que ha hecho que sea poco utilizado en la práctica.

Para afrontar este problema en el proyecto se han implementado diversos algoritmos (véase la sección 4.3.1) destacando entre ellos (por la velocidad de convergencia a la función objetivo y carga computacional), el algoritmo Baum-Welch Extendido (BWE). El entrenamiento generativo ML aun se ha mantenido para inicializar los modelos de idioma (representados por una mezcla de gaussianas o GMM), para que el entrenamiento posterior con BWE convergiera en lo mínimo posible a máximos locales menos óptimos.

El sistema implementado basado en BWE ha sido utilizado con éxito en las evaluaciones de idioma de NIST 2003, 2005 y en la evaluación de verificación de la lengua Albayzin 2008. Durante los experimentos y evaluaciones a las que ha sido sometido el sistema GMM-MMI, ha obtenido una mejora relativa de aproximadamente el 50 % sobre el reconocedor GMM-ML del mismo número de mezclas.

Otro rasgo importante del entrenamiento MMI es que emplea segmentos de audio en lugar de vectores individuales, como ocurre con el entrenamiento ML, añadiendo información característica de la segmentación particular de cada idioma (esto ha demostrado ser especialmente útil para mejorar la precisión en las evaluaciones de idioma NIST 2003 y 2005). La segmentación fue implementada con un reconocedor fonético de Húngaro, en donde todo el audio contendió entre silencios era considerado un segmento si tenía una duración superior a 0.5 segundos.

Después de completar con éxito la implementación del sistema GMM-MMI, se sintió la necesidad de compararlo con otros clasificadores discriminativos. Para ello se implementó también el sistema que se dio en llamar AGMM-SVM (véase la sección 2.4.2.2.3) consiguiendo muy buenos resultados en la evaluación de idioma de Albayzin 2008 (de hecho, el grupo ATVS fue el ganador de dicha evaluación).

Por último, se enumeran las posibles líneas de trabajo futuras:

- Reutilización del código del algoritmo Baum-Welch Extendido para la investigación y creación de clasificadores de reconocimiento automático en voz espontánea y reconocimiento automático de escritura.
- Fusión con otros sistemas de idioma del grupo ATVS. La fusión de sistemas es ampliamente usada en reconocimiento de idioma. Típicamente un sistema final estará compuesto de diversos sistemas acústicos fusionados con sistemas fonéticos (para que la información se complemente). Disponer de un nuevo clasificador permitirá probar con más combinaciones óptimas.
- Utilización del sistema GMM-MMI en conjunción con técnicas avanzadas de canal, como Joint Factor Analysis y Eigenchannel Compensation, en lugar de los clasificadores generativos GMM-UBM que se vienen usando.
- Optimización computacional de la implementación del algoritmo BWE. Como se vio en la sección 2.4.2.1.2, utilizando un UBM es posible realizar el entrenamiento MMI únicamente sobre las c mezclas más pesadas y sin que esto impacte en la optimización de la función objetivo MMI. Si D es la cantidad de audio por idioma, L el número de idiomas, M el número de mezclas de los GMM, entonces la implementación actual tendrá una complejidad $O(L*L*D*M)$ frente a $O(L*L*D*c)$ que tendría la nueva implementación.

Glosario

Curva DET Trazo gráfico de las tasas de error medidas. Por lo general, las curvas DET trazan las tasas de error de decisión (tasa de falso rechazo vs. tasa de falsa aceptación)

EER (Tasa de igual error), estadística utilizada para mostrar el rendimiento biométrico; por lo general, durante la tarea de verificación. La tasa EER es la ubicación en una curva ROC o DET donde la tasa de falsa aceptación y la tasa de falso rechazo son iguales. Por lo general, cuánto más bajo sea el valor de la tasa de igual error, mayor será la precisión del sistema biométrico. Observe, sin embargo, que la mayoría de los sistemas operativos no están preparados para funcionar con la tasa de igual error, de modo que la verdadera utilidad de esta medida está limitada a la comparación con el rendimiento del sistema biométrico

FAR Tasa de falsa aceptación, estadística utilizada para medir el rendimiento biométrico durante la tarea de verificación. Porcentaje de veces que un sistema produce una falsa aceptación, lo cual ocurre cuando un individuo es erróneamente vinculado con la información biométrica existente de otra persona

FRR (Tasa de falso rechazo),estadística utilizada para medir el rendimiento biométrico durante la tarea de verificación. Porcentaje de veces que el sistema produce un falso rechazo. Ocurre un falso rechazo cuando un individuo no es vinculado con su propia plantilla biométrica existente

GMM Gaussian Mixture Model (Modelo de Mezclas de Gausianas)

HMM (Hidden Markov Model), modelo oculto de Markov

HTK Toolkit de creación y tratamiento de modelos HMM diseñado por la universidad de Cambridge (CUED)

LID (Recocimiento de idioma), tecnología empleada en el indexado de contenidos multimedia, enrutamiento en servicios de atención telefónica y configuración de sistema. Consiste en determinar el idioma de los interlocutores

MAP (Maximum a Posteriori),método de adaptación de modelos independientes de locutor a los distintos locutores

MFCC (Mel Frequency Cepstral Coefficients),coeficientes cepstrales en escala de frecuencias Mel

NIST (National Institute of Standards and Technology),organismo federal, no regulador, perteneciente a la Cámara de Comercio de los Estados Unidos que desarrolla y promueve medidas, estándares y tecnología para aumentar la productividad, facilitar el comercio y mejorar la calidad de vida

Phone-SVM Sistema de reconocimiento de idioma que es semejante a un PPRLM pero que usa para extraer las puntuaciones un sistema SVMs sobre los n-gramas identificados

PPR (Parallel Phone Recognition), sistema de reconocimiento de idioma que además de usar un transcriptor fonético tiene un modelo del idioma a reconocer

PPRLM (Parallel PRLM), reconocimiento de idioma mediante modelos fonéticos de los mismos

PRLM (Phone Recognition followed by Language Modelling),sistema de reconocimiento de idioma construido con varios PRLM en paralelo

ROC (Característica de funcionamiento del receptor), método para mostrar el rendimiento de precisión medida de un sistema biométrico. La característica ROC en una verificación compara la tasa de falsa aceptación con la tasa de verificación

Sphinx	Toolkit de tratamiento de voz diseñado por la universidad Carnegie-Mellon
SVMs	(Support Vector Machines),método de reconocimiento de patrones

Bibliografía

[1] M. A. Zissman. Comparison of four approaches to automatic language identification of telephone speech. IEEE Transactions on Speech and Audio Processing, vol. 4, 1996.

[2] M. A. Zissman and K. M. Berkling. Automatic language identification. Speech Communication, 35 (1-2):115-124, Aug 2001.

[3] D. A. Reynolds. A Gaussian mixture modeling approach to text independent Speaker identification.Ph.D. thesis,Georgia Inst. of Technol., 1992.

[4] Reynold D. A., Quatieri T. F., Dunn R. B., (2000), Speaker Verification Using Adapted Gaussian Mixture Models, Digital Signal Processing, vol. 10, pp. 19-41.

[5] C. Corredor Ardoy, J.L. Gauvain, M. Adda-Decker, and L. Lamel. Language identification with language independent acoustic models. Proc. Eurospeech, vol. 1, pp. 5-8, Rhodes, Sep. 97.

[6] L. Lamel, J.L. Gauvain.Language identification using phone-based acoustic likelihoods. Proc. ICASSP, Adelaide, vol. 1, pp. 292-295, April 1994.

[7] T. J. Hazen and V. W. Zue. Automatic language identification using a segment-based approach. In Proceedings of Eurospeech 93, volume 2, pages 1303-1306, September 1993.

[8] J.A Swets. The relative operating characteristic in psychology. Science, vol. 182, pp. 990-1000, December 1973.

[9] J.L. Gauvain, A. Messaoudi, and H. Schwenk.Language recognition using phoneme lattices.In Proc. International Conferences on Spoken Language Processing (ICSLP), Sept. 2004, pp. 1283-1286.

[10] S.B.Davis and P.Mermelstein (1980). Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences.IEEE Transactions on Acoustics, Speech, and Signal Processing, 28(4), pp. 357-366.

[11] Felicity, A., Ambikairajah, E. In Warped magnitude and phase-based features for language identification,pp. 201 - 204,2006.

[12] J. Pelecanos and S. Sridharan,Feature warping for robust speaker verification, in Proc. ISCA Workshop on Speaker Recognition - 2001: A Speaker Oddyssey, June 2001.

[13] J. Pelecanos and S. Sridharan. Feature warping for robust speaker verification. in Proc. A Speaker Odyssey, The Speaker Recognition Workshop, 2001, pp 243-248.

[14] E. Wong and S. Sridharan. Methods to improve gaussian mixture model based language identification system, Proc. ICSLP, 2002, pp. 93-96.

[15] E. Wong and S. Sridharan, Fusion of Output Scores on Language. Identification System, Workshop on Multilingual Speech and Language.

[16] P. A. Torres-Carrasquillo, D. A. Reynolds, and J. R. Deller Jr., Language identification using Gaussian Mixture Model Tokenization, In ICASSP, Orlando, Fl., USA, 2002.

[17] Torres-Carrasquillo, P. A., E. Singer, et al. Approaches to language identification using Gaussian Mixture Models and the Shifted Delta Cepstrum. ICSLP, 2002.

[18] B. Bielefeld. Language identification using shifted delta cepstrum. In Fourteenth Annual Speech Research Symposium, 1994.

[19] Allen, F. R., E. Ambikairajah, et al. Language Identification using Warping and the Shifted Delta Cepstrum. IEEE International Workshop on Multimedia Signal Processing, Shanghai, China, 2005.

[20] Y. K. Muthusamy, R. A. Cole, and B. T. Oshika. The OGI multi-language telephone speech corpus. In Proc. ICSLP, pages 895-898, 1992.

[21] S. J. Young et al. The HTK Book. Cambridge University Engineering Department, 2001. (http://htk.eng.cam.ac.uk/docs/docs.shtml).

[22] W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R Leek. High-level speaker verification with support vector machines, in ICASSP Conference, Montreal, CANADA, May 2004, pp. 73-76.

[23] Roongroj Nopsuwanchai. Discriminative training methods and their applications to handwriting recognition. November 2005.

[24] Discriminative training for speech recognition. Erik McDermott March, 1997

[25] Discriminative Methods in HMM-based speech recognition - Valtchev - 1995

[26] L. Burqet, P. Metejka, and J. Cernocky.Discriminative training techniques for acoustic language identification. Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, pp.209-212, May 2006.

[27] C. White, I. Shafran, and J. Gauvain.Discriminative classifiers for language recognition. Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, pp.213-216, May 2006.

[28] Q. Dan and W. Bingxi.Discriminative training of gmm for language identication, in ISCA and IEEE Workshop on Spontaneous Speech Processing and Recognition(SSPR), Tokyo,Japan, Apr. 2003, p. MAP8.

[29] D. Povey.Discriminative Training for Large Vocabulary Speech Recognition. Ph.D. thesis, Cambridge University, July 2004.

[30] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance.In Proc. of the European Conference on Speech Communication and Technology (EUROSPEECH), Rhodes, Greece, September 1997, pp. 1895-8.

[31] CallFriend Corpus: http://www.ldc.upenn.edu/catalog.

[32] Kalaka, Speech database created for the 2008 Language Recognition Evaluation on Spanish Languages, organized by the Spanish Network on Speech Technology. Produced by the Software Technologies Working Group (GTTS, http://gtts.ehu.es), University of the Basque Country.

[33] NIST 2003 Language Recognition Evaluation website: http://www.nist.gov/speech/tests/lang/2003/

[34] NIST 2005 Language Recognition Evaluation website: http://www.nist.gov/speech/tests/lang/2005/

[35] Minghui Liu, Yanlu Xie, Zhiqiang Yao, Beiqian Dai.A New Hybrid GMM/SVM for Speaker Verification.icpr,pp.314-317, 18th International Conference on Pattern Recognition (ICPR'06) Volume 4, 2006.

[36] Wade Shen and Douglas Reynolds.Improved GMM-based language recognition using constrained MLLR transforms.

[37] L. Burget, P. Matejka, P. Schwarz, O. Glembek, and J. Cernocky .Analysis of feature extraction and channel compensation in GMM speaker recognition system. IEEE Transactions on Audio, Speech and Language Processing, vol. 15, no. 7, pp. 1979-1986, Sept. 2007.

[38] V. Hubeika, L. Burget, P. Matjka and P. Schwarz. Discriminative Training and Channel Compensation forcoustic Language Recognition, in Proc. Interspeech 2008.

[39] F. Castaldo, E. Dalmasso, P. Laface, D. Colibro, and C. Vair.Language identification using acoustic models and speaker compensated cepstral-time matrices.In Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP), Honolulu, Hawaii, USA, Oct. 2007, vol. 4, pp. 1013-1016.

[40] V. Hubeika, L. Burget, P. Matejka, and J. Cernocky.Channel compensation for speaker recognition. in Proc. Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms, June 2007.

[41] P.A. Torres-Carrasquillo, D. Sturim, D. Reynolds, and A. McCree. Eigenchannel Compensation and Discriminatively Trained Gaussian Mixture Models for Dialect and Accent Recognition, In INTERSPEECH-2008, 723-726.

[42] Software NETLAB: http://www.ncrg.aston.ac.uk/netlab/index.php

[43] Software STK: http://speech.fit.vutbr.cz/en/software/hmm-toolkit-stk

A

Presupuesto

1) Ejecución Material

▪ Compra de ordenador personal (Software incluido)	2.000 €
▪ Alquiler de impresora láser durante 6 meses	260 €
▪ Material de oficina	150 €
▪ Total de ejecución material	2.400 €

2) Gastos generales

▪ sobre Ejecución Material	352 €

3) Beneficio Industrial

▪ sobre Ejecución Material	132 €

4) Honorarios Proyecto

▪ 1800 horas a 15 € / hora	27000 €

5) Material fungible

▪ Gastos de impresión	280 €
▪ Encuadernación	200 €

6) Subtotal del presupuesto

▪ Subtotal Presupuesto	32.774 €

7) I.V.A. aplicable

▪ 16 % Subtotal Presupuesto	5.243,8 €

8) Total presupuesto

▪ Total Presupuesto	38.017,8 €

Madrid, Junio de 2011

El Ingeniero Jefe de Proyecto

Fdo.: Juan Bonillo Molina

Ingeniero Superior de Telecomunicación

B

Pliego de condiciones

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un *Reconocedor de idioma en voz espontanea mediante entrenamiento discriminativo MMI*. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales.

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad ”Presupuesto de Ejecución de Contrataz anteriormente llamado ”Presupuesto de Ejecución Material”que hoy designa otro concepto.

Condiciones particulares.

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.